On the limitations of the univariate marginal distribution algorithm to deception and where bivariate EDAs might help

Lehre, Per Kristian; Nguyen, Phan Trung Hai

DOI:
10.1145/3299904.3340316

License:
Other (please specify with Rights Statement)

Document Version
Peer reviewed version

Citation for published version (Harvard):
Lehre, PK & Nguyen, PTH 2019, On the limitations of the univariate marginal distribution algorithm to deception and where bivariate EDAs might help. in *Proceedings of the 15th ACM/SIGEVO Conference on Foundations of Genetic Algorithms (FOGA '19).* Association for Computing Machinery (ACM), New York, NY, USA, pp. 154-168, 15th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA XV), Potsdam, Germany, 27/08/19. https://doi.org/10.1145/3299904.3340316

On the Limitations of the Univariate Marginal Distribution Algorithm to Deception and Where Bivariate EDAs might help*

Per Kristian Lehre & Phan Trung Hai Nguyen
School of Computer Science
University of Birmingham
Birmingham B15 2TT, United Kingdom

July 30, 2019

Abstract

We introduce a new benchmark problem called Deceptive Leading Blocks (DLB) to rigorously study the runtime of the Univariate Marginal Distribution Algorithm (UMDA) in the presence of epistasis and deception. We show that simple Evolutionary Algorithms (EAs) outperform the UMDA unless the selective pressure μ/λ is extremely high, where μ and λ are the parent and offspring population sizes, respectively. More precisely, we show that the UMDA with a parent population size of $\mu = \Omega(\log n)$ has an expected runtime of $e^{\Omega(\mu)}$ on the DLB problem assuming any selective pressure $\frac{\mu}{\lambda} \geq \frac{14}{1000}$, as opposed to the expected runtime of $\mathcal{O}\left(n\lambda \log \lambda + n^3\right)$ for the non-elitist (μ, λ) EA with $\mu/\lambda \leq 1/e$. These results illustrate inherent limitations of univariate EDAs against deception and epistasis, which are common characteristics of real-world problems. In contrast, empirical evidence reveals the efficiency of the bi-variate MIMIC algorithm on the DLB problem. Our results suggest that one should consider EDAs with more complex probabilistic models when optimising problems with some degree of epistasis and deception.

*Preliminary version of this work will appear in the Proceedings of 15th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA XV), Potsdam, Germany

1 Introduction

Estimation of distribution algorithms (EDAs) [42, 43, 33] are a class of randomised search heuristics with many real-world applications (see [27] and references therein). Unlike traditional EAs, which define *implicit* models of promising solutions via genetic operations such as crossover and mutation, EDAs optimise objective functions by constructing and sampling *explicit* probabilistic models to generate *offspring* for the next iteration. The workflow of EDAs is an iterative process, where the initial model is a uniform distribution over the search space. The starting population consists of λ *individuals* sampled from the uniform distribution. A fitness function then scores each individual, and the algorithm selects the μ fittest individuals to update the model (where $\mu < \lambda$). The procedure is repeated until some termination condition is fulfilled, which is usually a threshold on the number of iterations or on the quality of the fittest offspring [27, 19].

Many variants of EDAs have been proposed over the last decades. They differ in the way their models are represented, updated as well as sampled over iterations. In general, EDAs are categorised into two main classes: *univariate* and *multivariate*. Univariate EDAs take advantage of first-order statistics (i.e. the mean) to build a probability vector-based model and assume independence between decision variables. The probabilistic model is represented as an n-vector, where each component is called a *marginal* (also *frequency*) and n is the problem instance size. Typical univariate EDAs are compact Genetic Algorithm (cGA [25]), Univariate Marginal Distribution Algorithm (UMDA [42]) and Population-Based Incremental Learning (PBIL [3]). In contrast, multivariate EDAs apply higher-order statistics to model the correlations between decision variables of the addressed problems.

The cGA is the simplest EDA, which operates on a population of two individuals and updates the probabilistic model additively via a parameter K, which is often referred to as the *hypothetical population size* of a genetic algorithm that the cGA is supposed to model. The two individuals are compared in terms of fitness to find the winner, and an increase of $\pm 1/K$ takes place at bit positions where individuals have different values. The algorithm also restricts the marginals to be within an interval $[1/n, 1 - 1/n]$, where the values $1/n$ and $1 - 1/n$ are called the *lower* and *upper borders* (or

margins), respectively, in order to prevent the marginals from fixing at trivial borders, which may cause the algorithm to converge prematurely. Such an algorithm is referred to as a cGA with margins. In contrast, the UMDA is another univariate EDA that has a larger population of λ individuals. In each so-called iteration, the marginal is renewed/updated to the frequency of 1-bit among the μ fittest individuals at each bit position (also called *empirical frequency*). Unlike the cGA, whose marginals can only get increased by an amount of $1/K$, those of the UMDA might jump between the upper and lower borders. A generalisation of the UMDA is the PBIL, where each marginal is updated following a convex combination of the current marginal and the empirical frequency via a so-called *smoothing parameter*. The marginals cannot change by a large amount, and it is then less likely that *genetic drift* [47] happens soon after the algorithm starts.

The theory of evolutionary computation literature provides rigorous analyses giving insights into the *runtime* (synonymously, *optimisation time*), that is the number of function evaluations of the studied algorithm until an optimal solution is sampled for the first time. In other words, theoretical work usually addresses the unlimited case when we consider the run of the algorithm as an infinite process. These analyses provide *performance guarantees* of the algorithm for a wide range of problem instance size.

Although EDAs were introduced several decades ago and have since shown strong potential as a global optimiser via many practical applications [27], the theoretical understanding of EDAs is very limited. There had been only a handful of runtime analyses of EDAs by 2015. Recently, they have drawn more attention from the community [11, 22, 32, 12, 47, 53, 51, 23, 36, 8, 37, 52, 40, 15, 26]. While rigorous runtime analyses provide deep insights into the performance of randomised search heuristics, it is highly challenging even for simple algorithms on toy functions. Most current runtime results merely concern univariate EDAs on functions like OneMax [32, 51, 36, 53, 40], LeadingOnes [15, 22, 37, 53, 38], BinVal [52, 37] and Jump [26, 11, 12], hoping that this provides valuable insights into the development of new techniques for analysing multivariate variants of EDAs and the behaviour of such algorithms on easy parts of more complex problem spaces [13]. There are two main reasons accounted for this. Firstly, the working principle of these algorithms, which are not originally designed to support the theoretical

analyses, is complicated, involving lots of randomnesses, and the interplay of decision variables usually has a huge impact on the overall runtime [13, 46]. Secondly, we are lacking in the state-of-the-art tools in algorithmics [13]. There are seemingly two currently popular techniques used to analyse EDAs. The first tool is *drift theorems* [28, 16, 31], while the other tool is the *level-based theorem*, first proposed in [35] and constantly improved upon [6, 14], in the context of non-elitist population-based EAs.

The fact that the UMDA never attempts to learn variable interactions leads some to conjecture that the algorithm will not perform well in environments with *epistasis* and *deception*. More specifically, epistasis corresponds to the maximum number of other variables each of the n decision variables depends on [9, 24]. We can take as an example the LeadingOnes function, which has a maximum epistasis level of n and is often used to study the ability of EAs to cope with variable dependency. Previous studies [7, 37, 38] show an $\mathcal{O}\left(n^2\right)$ expected runtime for the UMDA and the PBIL on this function, which seems to contradict the above-mentioned claim. However, Lehre and Nguyen [38] recently showed that univariate EDAs based on probability vectors have certain limitations to epistasis: if the *selective pressure* is larger than roughly $1/e$ (as previously required in [7, 37]) and the parent population is sufficiently large, the UMDA fails to optimise the LeadingOnes function in polynomial expected runtime.

Regarding deception, an example was already mentioned in [27], where the UMDA gets stuck in the concatenated trap of order 5 (called Trap-5 [1], where the original trap function is applied to each of $\lfloor n/5 \rfloor$ blocks of five consecutive bits) since marginals are deceived to hit the lower border. This excellent example demonstrates the limitations of the univariate model as statistics of low order is misleading [27]. However, this function might be as difficult for the UMDA as it is for the EAs [24]. We believe not only will the UMDA fail on this function, but it will also fail on some other problem with a milder degree of deception. To this end, a function where the UMDA takes an exponential expected runtime, while simple EAs has a polynomial expected runtime is still missing[1]. On the other hand, a function where the UMDA outperforms simple EAs does exist. Chen et al. [5] proposed the Substring function to point out the advantage of the probability vector-

[1]We define the terms "polynomial" and "exponential" as $n^{\mathcal{O}(1)}$ and $2^{n^{\Omega(1)}}$, respectively.

based model. More specifically, the $(1+1)$ EA needs a runtime of $2^{\Omega(n)}$ with probability $1-2^{-\Omega(n)}$, whereas the UMDA with $\lambda = \Omega(n^{2+\varepsilon})$, for any constant $\varepsilon > 0$, and $\mu = \lambda/2$ optimises the function in polynomial runtime with probability $1-2^{-\Omega(n)}$. We note that this result is very limited in many senses that the population size is large, the selective pressure is fixed to $1/2$ and the considered UMDA does not have borders.

Motivated by this, we introduce a new benchmark problem which has a maximum epistasis level of n and is mildly deceptive. The fitness depends on the number of leading 11s, and reaching a unique global optimum requires overcoming many mild traps. Generally speaking, this function is harder than the LeadingOnes function, but still much easier compared to the Trap-5 function. The problem, which we call *Deceptive Leading Blocks* (DLB), can be formally defined over a finite binary search space $\mathcal{X} \coloneqq \{0,1\}^n$ as follows.

$$\mathrm{DLB}(x) \coloneqq \begin{cases} n & \text{if } \phi(x) = n/2, \\ 2 \cdot \phi(x) + 1 & \text{if } x_{2\phi(x)+1} + x_{2\phi(x)+2} = 0, \\ 2 \cdot \phi(x) & \text{if } x_{2\phi(x)+1} + x_{2\phi(x)+2} = 1, \end{cases}$$

where

$$\phi(x) \coloneqq \sum_{i=1}^{n/2} \prod_{j=1}^{2i} x_j \tag{1}$$

denotes the number of leading 11s, which is identical to $\mathrm{LOB}_2(x)$ in [29, Definition 13]. The global optimum is the all-ones bitstring. The bitstring is partitioned into independent blocks of $w \geq 2$ consecutive bits. The difficulty of the problem is determined by the width w of each block, that can be altered to increase the level of deception. Here, we consider the smallest width $w = 2$, as this suffices to prove an exponential gap between the runtimes of the UMDA and the (μ,λ) EA. DLB is similar to the LeadingOnes function except it attempts to deceive the algorithm by assigning tricky weights to different settings of the leftmost non-11 block, which we call the *active block*. Each leading 11 contributes a value of two, while a value of one is awarded if the active block is a 00, and no reward is given for any other blocks. The all-ones bitstring has a fitness of $(n/2) \cdot 2 = n$ since there are $n/2$ blocks, assuming that n is a multiple of $w = 2$.

Table 1: Expected runtime of some simple EAs and the UMDA (with borders) on the DLB function.

Algorithm	Sel. Pressure	Pop. Size	Expected Runtime
$(1+\lambda)$ EA	–	–	$\mathcal{O}(n\lambda + n^3)$
$(\mu+1)$ EA	–	–	$\mathcal{O}(\mu n \log n + n^3)$
(μ,λ) EA	$\mu/\lambda = \mathcal{O}(1)$	$\lambda = \Omega(\log n)$	$\mathcal{O}(n\lambda \log \lambda + n^3)$
UMDA	$\mu/\lambda = \mathcal{O}(1/\mu)$	$\mu = \Omega(\log n)$	$\mathcal{O}(n\lambda \log \lambda + n^3)$
	$\mu/\lambda > \frac{14}{1000}$	$\mu = \Omega(\log n)$	$e^{\Omega(\mu)}$

By studying this problem, we first show that simple EAs, including elitist and non-elitist variants like $(1+\lambda)$ EA, $(\mu+1)$ EA and (μ,λ) EA, and some Genetic Algorithms, can optimise the DLB function within an $\mathcal{O}\left(n^3\right)$ expected runtime. We then show that the UMDA fails to optimise this function in polynomial expected runtime, assuming that the selective pressure is $\mu/\lambda \geq \frac{14}{1000}$, and the parent population size μ is sufficiently large. More specifically, the expected runtime is $n^{\Omega(1)}$ when $\mu = \Theta(\log n)$, while a lower bound of $2^{\Omega(n^\varepsilon)}$ is achieved for $\mu \geq cn^\varepsilon$ for some constants $c, \varepsilon > 0$. In the latter case, we say the UMDA is fooled by deceptive fitness. On the other hand, if the selective pressure is in the order of $1/\mu$ (i.e., extremely high), we obtain an upper bound of $\mathcal{O}\left(n^3 + n\lambda \log \lambda\right)$ on the expected runtime of the UMDA on the DLB function. Intuitively speaking, under this extreme selective pressure the UMDA selects few fittest individuals to update the probabilistic model, and we believe (with some empirical evidence in Section 5) that the UMDA degenerates into the $(1,\lambda)$ EA, where only the fittest offspring is selected to the next generation. Table 1 summarises the main results in this paper.

Last but not least, many algorithms similar to the UMDA with fitness proportional selection have a wide range of applications in *bioinformatics* [2]. The algorithms relate to the notion of *linkage equilibrium* [45] – a popular model assumption in population genetics. Studying the UMDA might solidify our understanding of population dynamics. Based on results from this paper, we believe that the UMDA and other univariate model-based algorithms must be tuned carefully when optimising objective functions

with epistasis and might not perform well in deceptive environments.

The paper is structured as follows. Section 2 introduces the algorithms, including the UMDA, EAs and the MIMIC algorithm. Section 3 provides analyses of the expected runtime of simple EAs and GAs on the DLB function, followed by a detailed runtime analysis for the UMDA on the DLB function. Next, we illustrate the efficiency of the MIMIC algorithm on the DLB function in Section 5 via a small empirical study. Finally, Section 6 gives some concluding remarks and suggests future work.

2 Preliminaries

This section briefly describes the algorithms studied in this paper. Recall that $\mathcal{X} = \{0,1\}^n$. Each *individual* (or bitstring) is represented as $x = (x_1, x_2, \ldots, x_n) \in \mathcal{X}$. The *population* of λ individuals in an iteration $t \in \mathbb{N}$ is denoted as $P_t \coloneqq (x_t^{(1)}, \ldots, x_t^{(\lambda)})$. We consider in this paper the maximisation of an *objective function* $f : \mathcal{X} \to \mathbb{R}$. Denote $[n] \coloneqq \mathbb{N} \cap [1, n]$.

2.1 Evolutionary algorithms

The $(\mu + \lambda)$ EA is a mutation-only EA, which operates on a population of μ individuals. In each iteration, the algorithm generates λ new offspring by flipping bits in each of λ individuals, chosen uniformly at random from the population, independently with mutation rate $1/n$. Afterwards, the μ fittest individuals out of a pool of $\lambda + \mu$ individuals are selected to form the new population. The algorithm is *elitist* since the best fitness discovered thus far is guaranteed to never decrease. Popular variants of the algorithm are (1+1) EA [17, Algorithm 1], (1+λ) EA [30, Algorithm 1] and (μ+1) EA [50, Definition 1]. For comparison, we also consider the non-elitist (μ, λ) EA, defined in Algorithm 1 [39], with mutation rate χ/n for any constant $\chi \in (0, n/2)$. In this algorithm, the next population consists of the μ fittest individuals chosen from a set of $\lambda > \mu$ offspring produced by mutation.

2.2 Univariate marginal distribution algorithm

The UMDA, defined in Algorithm 2, maintains a univariate model in each iteration $t \in \mathbb{N}$ that is represented as an n-vector $p_t \coloneqq (p_{t,1}, \ldots, p_{t,n})$, where

Algorithm 1: (μ, λ) EA with mutation rate χ/n

$t \leftarrow 0$
$P_t \leftarrow (x^{(1)}, x^{(2)}, \ldots, x^{(\mu)})$ uniformly at random from $\mathcal{X}^\lambda$
repeat
  for $i = 1, 2, \ldots, \lambda$ **do**
    select $j \in [\mu]$ uniformly at random
    create $y^{(i)}$ by flipping each bit in $x^{(j)}$ independently with probability χ/n
  sort $(y^{(1)}, \ldots, y^{(\lambda)})$ such that $f(y^{(1)}) \geq \ldots \geq f(y^{(\lambda)})$, where ties are broken uniformly at random
  $P_{t+1} \leftarrow (y^{(1)}, y^{(2)}, \ldots, y^{(\mu)})$
  $t \leftarrow t + 1$
until *termination condition is fulfilled*

each marginal (or frequency) $p_{t,i}$ for each $i \in [n]$ is the probability of sampling a 1 at the i-th bit position of the offspring. The probability of sampling $x = (x_1, x_2, \ldots, x_n)$ from the model p_t is

$$\Pr(x \mid p_t) = \prod_{i=1}^{n} (p_{t,i})^{x_i} (1 - p_{t,i})^{1-x_i}.$$

The starting model is the uniform distribution $p_0 \coloneqq (1/2, \ldots, 1/2)$. In each so-called iteration, the algorithm samples a population P_t of λ individuals and sorts them in descending order according to fitness. Let $X_{t,i}$ denote the number of 1s in bit position $i \in [n]$ among the μ fittest individuals. The marginals are updated using the component-wise formula: $p_{t+1,i} \coloneqq X_{t,i}/\mu$ for all $i \in [n]$. Recall that $\gamma^* = \mu/\lambda \in (0, 1]$ is the *selective pressure* of the algorithm. The algorithm also restricts the marginals to be within $[1/n, 1 - 1/n]$ to avoid premature convergence.

2.3 Mutual-information-maximising input cluster algorithm

The MIMIC [10] is a well-known bivariate EDA, which takes advantage of second-order statistics in an attempt to model the correlations between decision variables. More specifically, let $p^*(X)$ denote the true distribution underlying the μ selected individuals, where $X = (X_1, X_2, \ldots, X_n)$. It is

Algorithm 2: UMDA with margins

1 $t \leftarrow 0$
2 initialise $p_t \leftarrow (1/2, 1/2, \ldots, 1/2)$
3 **repeat**
4 **for** $j = 1, 2, \ldots, \lambda$ **do**
5 sample $x_{t,i}^{(j)} \sim \text{Bernoulli}(p_{t,i})$ for each $i \in [n]$
6 sort $(x_t^{(1)}, \ldots, x_t^{(\lambda)})$ such that $f(x^{(1)}) \geq \ldots \geq f(x^{(\lambda)})$, where ties are broken uniformly at random
7 **for** $i = 1, 2, \ldots, n$ **do**
8 $X_{t,i} = \sum_{j=1}^{\mu} x_{t,i}^{(j)}$
9 $p_{t+1,i} \leftarrow \max\{1/n, \min\{1 - 1/n, X_{t,i}/\mu\}\}$
10 $t \leftarrow t + 1$
11 **until** *termination condition is fulfilled*

often intractable to learn the true distribution $p^*(X)$, so an approximation to the distribution is often preferred. Following this approach, the MIMIC approximates $p^*(X)$ by a chain-structured model $\hat{p}(X)$, which is easy to learn and sample. Given a permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ of a set $[n]$, the chain-structured model is defined as follows:

$$\hat{p}_\pi(X) = p(X_{\pi_1})p(X_{\pi_2} \mid X_{\pi_1}) \cdots p(X_{\pi_n} \mid X_{\pi_{n-1}}),$$

where the X_{π_1} is called the root of the chain. The parameters of the model are then derived by minimising the Kullback-Leibler divergence between models $\hat{p}_\pi(X)$ and $p^*(X)$, which is equivalent to minimising an alternative cost function

$$J_\pi(X) = h(X_{\pi_1}) + h(X_{\pi_2} \mid X_{\pi_1}) + \cdots + h(X_{\pi_n} \mid X_{\pi_{n-1}}),$$

where $h(X_{\pi_i}) = -\mathbb{E}[\log p(X_{\pi_i})]$ and $h(X_{\pi_i} \mid X_{\pi_j}) = -\mathbb{E}[\log p(X_{\pi_i} \mid X_{\pi_j})]$ are the entropy and conditional entropy, respectively [10].

The model is constructed in each iteration as follows. The variable with the smallest entropy is selected to become the root of the chain. The variable among the remaining variables that has the smallest conditional entropy on the root is then added to the root. This procedure is repeated until all variables are added to the chain (see steps 5–7). Sampling from the

Algorithm 3: MIMIC with margins, where ties occurring in sorting and selection are broken uniformly at random.

1 $t \leftarrow 0$

2 $P_t \leftarrow (x^{(1)}, x^{(2)}, \ldots, x^{(\lambda)})$ uniformly at random from $\mathcal{X}^\lambda$

3 **repeat**

4 sort P_t such that $f(x^{(1)}) \geq f(x^{(2)}) \geq \ldots \geq f(x^{(\lambda)})$

5 $\pi_1^{(t+1)} \leftarrow \operatorname{argmin}_{j \in [n]} h(X_j)$

6 **for** $k = 2, 3, \ldots, n$ **do**

7 $\pi_k^{(t+1)} \leftarrow \operatorname{argmin}_j h(X_j \mid X_{\pi_{k-1}^{(t+1)}})$ where $j \in [n] \setminus \{\pi_1^{(t+1)}, \ldots, \pi_{k-1}^{(t+1)}\}$

8 **for** $j = 1, 2, \ldots, \lambda$ **do**

9 $x^{(j)}_{\pi_1^{(t+1)}} \leftarrow \begin{cases} 1 & \text{w.p.} \quad \mathrm{R}(p(X_{\pi_1^{(t+1)}})), \\ 0 & \text{w.p.} \quad 1 - \mathrm{R}(p(X_{\pi_1^{(t+1)}})) \end{cases}$

10 **for** $k = 2, 3, \ldots, n$ **do**

11 $x^{(j)}_{\pi_k^{(t+1)}} \leftarrow \begin{cases} 1 & \text{w.p.} \quad \mathrm{R}(p(X_{\pi_k^{(t+1)}} \mid X_{\pi_{k-1}^{(t+1)}})), \\ 0 & \text{w.p.} \quad 1 - \mathrm{R}(p(X_{\pi_k^{(t+1)}} \mid X_{\pi_{k-1}^{(t+1)}})) \end{cases}$

12 $P_{t+1} \leftarrow (x^{(1)}, x^{(2)}, \ldots, x^{(\lambda)})$

13 $t \leftarrow t + 1$

14 **until** *termination condition is fulfilled*

chain-structured model is even simpler. The root is sampled first using its marginal probability. The next variable in the chain is then sampled using the conditional probability on the preceding variable. This is repeated until the end of the chain is reached (see steps 8–11). The procedure is known as *ancestral sampling* [4]. Denote the permutation in an iteration $t \in \mathbb{N}$ as $\pi^{(t)} \coloneqq \{\pi_1^{(t)}, \pi_2^{(t)}, \ldots, \pi_n^{(t)}\}$, where $\pi_1^{(t)}$ denotes the root.

We note that the description of the MIMIC algorithm in [10] is very ambiguous. In particular, when constructing the model, we need to calculate many conditional entropies, which in turn require the calculation of many conditional probabilities. We can take as an example the conditional probability $p(X_{\pi_i} = \alpha \mid X_{\pi_j} = \beta)$ for $\alpha, \beta \in \{0, 1\}$, which by the definition can be

Algorithm 4: Non-elitist population-based algorithm

```
t ← 0; create initial population P_t
repeat
    for i = 1, ..., λ do
        sample P_{t+1,i} ~ D(P_t)
    t ← t + 1
until termination condition is fulfilled
```

written as

$$p(X_{\pi_i} = \alpha \mid X_{\pi_j} = \beta) = p(X_{\pi_i} = \alpha, X_{\pi_j} = \beta)/p(X_{\pi_j} = \beta).$$

Note that all probabilities on the right-hand side will be directly measured from the sampled population (via counting). And, this is where the problem comes from since very often that the event $\{X_{\pi_j} = \beta\}$ may not happen, leading to the probability estimate $p(X_{\pi_j} = \beta) \approx 0$ which renders the conditional probability above undefined. To avoid this problem, we make use of a function $R(p) \coloneqq \max\{p, 1/n\}$ to ensure that the probability $p(X_{\pi_j} = \beta)$ is always positive, and never less than $1/n$. Note that this bound is in line with the way marginal probabilities are bounded in other EDAs, such as the UMDA. We think that this small change is essential to ensure that the pseudo-code of MIMIC is well-defined. Algorithm 3 gives a full description of the MIMIC (with margins) [44, Chapter 13].

2.4 Level-based analysis

First proposed in [35], the level-based theorem is a general tool that provides upper bounds on the expected runtime of many non-elitist population-based algorithms on a wide range of optimisation problems [6, 14, 7, 8, 36, 37, 38]. The theorem assumes that the studied algorithm can be described in the form of Algorithm 4, which never assumes specific fitness functions, selection mechanisms, or generic operators like mutation and crossover. The search space $\mathcal{X}$ is partitioned into m disjoint subsets $A_1, \ldots, A_m$, which we call *levels*, and the last level A_m consists of global optima of the objective function. Denote $A_{\geq j} \coloneqq \cup_{k=j}^{m} A_k$.

Theorem 1 ([6]). *Given a partition $(A_i)_{i\in[m]}$ of $\mathcal{X}$, define $T \coloneqq \min\{t\lambda \mid |P_t \cap A_m| > 0\}$, where for all $t \in \mathbb{N}$, $P_t \in \mathcal{X}^\lambda$ is the population of Algorithm 4 in iteration t. Denote $y \sim \mathcal{D}(P_t)$. If there exist $z_1, \ldots, z_{m-1}, \delta \in (0,1]$, and $\gamma_0 \in (0,1)$ such that for any population $P_t \in \mathcal{X}^\lambda$,*

(G1) for each level $j \in [m-1]$, if $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda$ then

$$\Pr\left(y \in A_{\geq j+1}\right) \geq z_j,$$

(G2) for each level $j \in [m-2]$ and all $\gamma \in (0, \gamma_0]$, if $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda$ and $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda$ then

$$\Pr\left(y \in A_{\geq j+1}\right) \geq (1+\delta)\,\gamma,$$

(G3) and the population size $\lambda \in \mathbb{N}$ satisfies

$$\lambda \geq \left(\frac{4}{\gamma_0\delta^2}\right) \ln\left(\frac{128m}{z_*\delta^2}\right),$$

where $z_* \coloneqq \min_{j\in[m-1]}\{z_j\}$, then

$$\mathbb{E}\left[T\right] \leq \left(\frac{8}{\delta^2}\right) \sum_{j=1}^{m-1} \left[\lambda \ln\left(\frac{6\delta\lambda}{4 + z_j\delta\lambda}\right) + \frac{1}{z_j}\right].$$

2.5 Useful Tools from Probability Theory

We will use the following well-known tail bounds [41, 18].

Lemma 1 (Chernoff Bound). *Let $b > 0$. Let $Y_1, \ldots, Y_n$ be independent random variables (not necessarily i.i.d.), that take values in $[0, b]$. Let $Y \coloneqq \sum_{i=1}^n Y_i$, and $\mu \coloneqq \mathbb{E}[Y]$. Then for any $0 \leq \delta \leq 1$,*

$$\Pr(Y \leq (1-\delta)\mu) \leq e^{-\delta^2\mu/(2b)},$$

and

$$\Pr(Y \geq (1+\delta)\mu) \leq e^{-\delta^2\mu/(3b)}.$$

Lemma 2 (Chernoff-Hoeffding Bound). *Let $Y_1, \ldots, Y_n$ be independent random variables (not necessarily i.i.d.), where Y_i takes values in $[0, b_i]$. Let $Y \coloneqq \sum_{i=1}^n Y_i$, and let $b \coloneqq \sum_{i=1}^n b_i^2$. Then $\Pr(|Y - \mathbb{E}[Y]| \geq t) \leq 2e^{-2t^2/b}$.*

Lemma 3 ([49]). *$\mathbb{E}[X^2 \mid X \sim Bin\,(n,p)] = np(p(n-1)+1)$.*

3 EAs optimise DLB efficiently

We start by showing that simple EAs optimise the DLB function in polynomial expected runtime. We consider both elitist and non-elitist EAs, namely $(1+\lambda)$ EA, $(\mu+1)$ EA and (μ,λ) EA, in addition to Genetic Algorithms. Although these EAs are simple, we analyse them here to emphasise their efficiency in dealing with epistasis and mild deception. Speaking of proving techniques, we will use the fitness-level method [48] and the level-based theorem (see Theorem 1). In doing so, the search space $\mathcal{X}$ is first partitioned into non-empty disjoint subsets $A_0, A_1, \ldots, A_m$ (called *levels*, where $m \coloneqq n/2$) such that

$$A_i = \{x \in \mathcal{X} : \phi(x) = i\}, \tag{2}$$

where $\phi(x)$ is defined in (1), and A_m contains the all-ones bitstring. We now give runtime bounds on the DLB function for the EAs; the proofs are straightforward.

Theorem 2. *The expected runtime of the* $(1+\lambda)$ EA *on the* DLB *function is* $\mathcal{O}\left(\lambda n + n^3\right)$.

Proof. Levels are defined as in (2). The probability of leaving the current level $i < m$ is lower bounded by $(1-1/n)^{n-2}(1/n)^2 \geq 1/en^2$, and thus not leaving it happens with probability at most $1 - 1/en^2$. In each iteration, the $(1+\lambda)$ EA samples λ individuals by mutating the current bitstring. At least one among λ individuals leaves the current level with probability at least $1-(1-1/en^2)^\lambda \geq 1 - e^{-\lambda/en^2}$. Note that if $\lambda \geq en^2$, then this probability is at least $1-1/e$; otherwise, it is at least $\lambda/2en^2$. Putting everything together, the expected runtime guaranteed by the fitness-level method is

$$\lambda \cdot \sum_{i=0}^{n/2-1} \left(\mathcal{O}(1) + \frac{2en^2}{\lambda}\right) = \mathcal{O}\left(n\lambda + n^3\right). \qquad \square$$

Theorem 3. *The expected runtime of the* $(\mu+1)$ EA *on the* DLB *function is* $\mathcal{O}\left(\mu n \log n + n^3\right)$.

Proof. Levels are defined as in (2). It suffices to correct block $i+1$ to leave the current level. Following [50], we define a fraction $\chi(i) \coloneqq n/\log n$. Given j copies of the best individual, another one is created with probability

$(j/\mu)(1-1/n)^n \geq j/2e\mu$. Thus, the expected time for a fraction $\chi(i)$ of the population to be in level i is given by

$$T_0 \leq 2e\mu \sum_{j=1}^{n/\log n} (1/j) \leq 2e\mu \log n.$$

Now given $\chi(i)$ individuals in level i, the event of leaving this level occurs with probability $s_i \geq (\chi(i)/\mu)(1-1/n)^{2i}(1/n)^2 \geq (\chi(i)/\mu) \cdot 1/en^2$. This probability is at least

$$s_i = \begin{cases} 1/en^2, & \text{if } \mu \leq \chi(i) \\ 1/(e\mu \log n), & \text{if } \mu > \chi(i). \end{cases}$$

The expected runtime of the algorithm on DLB is

$$\sum_{i=0}^{n/2-1} \left(T_0 + \frac{1}{s_i} \right) = \mathcal{O}\left(\mu n \log n + n^3\right). \qquad \square$$

Theorem 4. *The expected runtime of the* (μ, λ) EA *with* $\lambda \geq c \log n$ *for some sufficiently large constant* $c > 0$ *and* $\mu \leq \lambda e^{-2\chi}/(1+\delta)$ *for any constant* $\delta > 0$ *and a mutation rate constant* $\chi \in (0, n/2)$ *on the* DLB *function is* $\mathcal{O}\left(n\lambda \log \lambda + n^3\right)$.

Proof. Since (μ, λ) EA is non-elitist, Theorem 1 guarantees an upper bound on the expected runtime as long as the three conditions (G1), (G2) and (G3) are fully verified. Choose $\gamma_0 := \mu/\lambda$. The levels are defined as in (2), and A_j is assumed to be the current level.

Condition (G1) requires a lower bound on the probability of sampling an offspring in $A_{\geq j+1}$, where $A_{\geq j+1} := \cup_{k=j+1}^{m} A_k$, given $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda = \mu$. During the selection step, if we choose an individual in $A_{\geq j}$, then the step is successful if the mutation operator correctly flips two of the bits in the active block while keeping others unchanged. Thus, the probability of a successful sampling is at least

$$(1-\chi/n)^{n-2}(\chi/n)^2 \geq e^{-\chi}\chi^2/n^2.$$

Next, condition (G2) assumes that at least $\gamma\lambda < \mu$ individuals have at least $j+1$ leading 11s. It suffices to pick one of the $\gamma\lambda$ fittest individuals and flip none of the bits; the probability is at least

$$(\gamma\lambda/\mu)(1-\chi/n)^n \geq (\gamma/\gamma_0)e^{-2\chi} \geq (1+\delta)\gamma$$

if $\gamma_0 \leq e^{-2\chi}/(1+\delta)$ for any constant $\delta > 0$.

Putting everything into condition (G3) yields $\lambda \geq c\log(n)$ for a sufficiently large constant $c > 0$. Having fully verified three conditions, the expected runtime of the (μ, λ) EA on DLB is

$$\mathcal{O}\left(\sum_{i=0}^{n/2-1}\left(\lambda\log\lambda + n^2\right)\right) = \mathcal{O}\left(n\lambda\log\lambda + n^3\right). \qquad \square$$

So far we have considered only mutation-based EAs, the following theorem shows that Genetic Algorithms, defined in [6, Algorithm 2], with crossover rate p_c using any crossover operator also take a polynomial expected runtime to optimise the DLB function.

Theorem 5. *Genetic Algorithms with crossover rate $p_c = 1-\Omega(1)$ using any crossover operator, the bitwise mutation operator with mutation rate χ/n for any fixed constant $\chi > 0$ and one of the following selection mechanisms: k-tournament selection, (μ, λ)-selection, linear or exponential ranking selection, with their parameters k, λ/μ and η being set to no less than $(1+\delta)e^{\chi}/(1-p_c)$ where $\delta \in (0, 1]$ being any constant, take an $\mathcal{O}\left(n^3 + n\lambda\log\lambda\right)$ expected runtime on the* DLB *function, where $\lambda \geq c\log n$ for some sufficiently large constant $c > 0$.*

Proof. The results for k-tournament, (μ, λ)-selection and linear ranking follow by applying [35, Lemmas 5–7], while the result for exponential ranking can be seen in [6, Lemma 3]. $\square$

4 Why is UMDA inefficient on DLB?

Before we get to analysing the UMDA on the DLB function, we introduce some notation. Recall that there are $m := n/2$ blocks. We then let $C_{t,i}$ for each $i \in [m]$ denote the number of individuals having at least i leading 11s in iteration $t \in \mathbb{N}$, and $D_{t,i}$ denote the number of individuals having $i-1$ leading 11s, followed by a 00 at the i-th block. For the special case of $i = 1$, $D_{t,1}$ consists of those with the first block being a 00. We also let $E_{t,i}$ denote the number of individuals having $i-1$ leading 11s, followed by a 10 at the i-th block, and again $E_{t,1}$ consists of those having the first block being a 10.

Once the population has been sampled, the algorithm invokes truncation selection to select the μ fittest individuals to update the probability vector. We take this μ-cutoff into account by defining a random variable

$$Z_t \coloneqq \max\{i \in \mathbb{N} \cap [0, m] : C_{t,i} \geq \mu\}, \tag{3}$$

which tells us how many consecutive marginals, counting from position one, are set to the upper border $1 - 1/n$ in iteration t. We also define another random variable

$$Z_t^* \coloneqq \max\{i \in \mathbb{N} \cap [0, m] : C_{t,i} > 0\} \tag{4}$$

to be the number of leading 11s of the fittest individual(s). For readability, we often leave out the indices of random variables like when we write C_t instead of $C_{t,i}$, if values of the indices are clear from the context. Furthermore, let $(\mathcal{F}_t)_{t\in\mathbb{N}}$ be a filtration induced from the population $(P_t)_{t\in\mathbb{N}}$, and we often write $\mathbb{E}_t[X] \coloneqq \mathbb{E}[X \mid \mathcal{F}_t]$ and $\mathrm{Var}_t[X] \coloneqq \mathrm{Var}[X \mid \mathcal{F}_t]$.

4.1 On the distributions of $C_{t,i}, D_{t,i}$ and $E_{t,i}$

We apply the *principle of deferred decisions* [41] and imagine that the algorithm first samples the values of the first block for λ individuals. Once this is finished, it moves on to the second block and so on until the whole population is obtained.

We note that *selection* prefers individuals with the first block being a 11 to those with a 00, which in turn is more preferred to those with a 10 or 01 (due to deceptive fitness). The number of 11s in the first block follows a binomial distribution with parameters λ and $p_{t,1}p_{t,2}$, that is, $C_{t,1} \sim \mathrm{Bin}\,(\lambda, p_{t,1}p_{t,2})$. Having sampled $C_{t,1}$ 11s, there are $\lambda - C_{t,1}$ other blocks in block 1 in the current population. $D_{t,1}$ is also binomially distributed with parameters $\lambda - C_{t,1}$ and $(1 - p_{t,1})(1 - p_{t,2})/(1 - p_{t,1}p_{t,2})$ by the definition of conditional probability since the event of sampling a 11 is excluded. Similarly having sampled 11s and 00s, $E_{t,1}$ is binomially distributed with $\lambda - C_{t,1} - D_{t,1}$ trials and success probability $(p_{t,1}(1 - p_{t,2}))/(1 - p_{t,1}p_{t,2} - (1 - p_{t,1})(1 - p_{t,2}))$ since again the event of sampling either a 11 or a 00 is excluded. Finally, the number of 01s is $\lambda - C_{t,1} - D_{t,1} - E_{t,1}$.

Having sampled the first block for λ individuals, and note that the *bias* due to selection in the second block comes into play only if the first block is

a 11. Among the $C_{t,1}$ fittest individuals, those with a 11 in the second block will be ranked first, followed by those with a 00, and finally with a 10 or 01. Conditioned on the first block being a 11, the number of 11s in the second block is binomially distributed with parameters $C_{t,1}$ and $p_{t,3}p_{t,4}$, i.e., $C_{t,2} \sim \mathrm{Bin}\,(C_{t,1}, p_{t,3}p_{t,4})$, and the number of 00s also follows a binomial distribution with $C_{t,1} - C_{t,2}$ trials and success probability $(1-p_{t,3})(1-p_{t,4})/(1-p_{t,3}p_{t,4})$. Similarly, $E_{t,2}$ is binomially distributed with parameters $C_{t,1} - C_{t,2} - D_{t,2}$ and $p_{t,3}(1-p_{t,4})/(1-p_{t,3}p_{t,4}-(1-p_{t,3})(1-p_{t,4}))$, and finally the number of 01s equals $C_{t,1} - C_{t,2} - D_{t,2} - E_{t,2}$. Unlike the first block, we also have $\lambda - C_{t,1}$ remaining individuals, and since there is *no bias* in the second block among these individuals, the numbers of 1s sampled at the two bit positions are binomially distributed with $\lambda - C_{t,1}$ trials and success probabilities $p_{t,3}$ and $p_{t,4}$, respectively.

We now consider an arbitrary block $i \in [m]$. By induction, we observe that the number of individuals having at least i leading 11s follows a binomial distribution with parameters $C_{t,i-1}$ and $p_{t,2i-1}p_{t,2i}$, that is,

$$C_{t,i} \sim \mathrm{Bin}\,(C_{t,i-1}, p_{t,2i-1}p_{t,2i})\,. \tag{5}$$

Similarly,

$$D_{t,i} \sim \mathrm{Bin}\left(C_{t,i-1} - C_{t,i}, \frac{(1-p_{t,2i-1})(1-p_{t,2i})}{1-p_{t,2i-1}p_{t,2i}}\right), \tag{6}$$

and

$$E_{i,t} \sim \mathrm{Bin}\left(C_{t,i-1} - C_{t,i} - D_{t,i}, \frac{p_{t,2i-1}(1-p_{t,2i})}{p_{t,2i-1}+p_{t,2i}-2p_{t,2i-1}p_{t,2i}}\right). \tag{7}$$

Finally, the number of individuals with $i-1$ leading 11s followed by a 01 in the block i is $C_{t,i-1} - C_{t,i} - D_{t,i} - E_{t,i}$. For the $\lambda - C_{t,i-1}$ remaining individuals, the numbers of 1s sampled in the bit positions $2i-1$ and $2i$ follow a $\mathrm{Bin}\,(\lambda - C_{t,i-1}, p_{t,2i-1})$ and a $\mathrm{Bin}\,(\lambda - C_{t,i-1}, p_{t,2i})$, respectively. We note in particular that by the end of this alternative view on the sampling process, we obtain the population of λ individuals sorted in descending order according to fitness, where ties are broken uniformly at random. The following lemma provides the expectations of these random variables.

Lemma 4. *For all $t \in \mathbb{N}$, if $i = 1$ then*

$$\begin{aligned}
\mathbb{E}_{t-1}[C_{t,i}] &= \lambda \cdot p_{t,2i-1}p_{t,2i}, \\
\mathbb{E}_{t-1}[D_{t,i}] &= \lambda \cdot (1 - p_{t,2i-1})(1 - p_{t,2i}), \\
\mathbb{E}_{t-1}[E_{t,i}] &= \lambda \cdot p_{t,2i-1}(1 - p_{t,2i}).
\end{aligned}$$

Otherwise, if $i \in [m] \setminus \{1\}$, then

$$\mathbb{E}_{t-1}[C_{t,i}] = \mathbb{E}_{t-1}[C_{t,i-1}] \cdot p_{t,2i-1}p_{t,2i}, \tag{8}$$

$$\mathbb{E}_{t-1}[D_{t,i}] = \mathbb{E}_{t-1}[C_{t,i-1}] \cdot (1 - p_{t,2i-1})(1 - p_{t,2i}), \tag{9}$$

$$\mathbb{E}_{t-1}[E_{t,i}] = \mathbb{E}_{t-1}[C_{t,i-1}] \cdot p_{t,2i-1}(1 - p_{t,2i}). \tag{10}$$

Proof. For the special case of $i = 1$, the expectations are trivial since random variables $C_{t,i}, D_{t,i}$ and $E_{t,i}$ are all binomially distributed with λ trials. In the remainder of the proof, we will consider the case of $i \neq 1$. By (5), the tower rule $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X \mid Y]]$ [21] and noting that p_t is $\mathcal{F}_{t-1}$-measurable, we get

$$\begin{aligned}
\mathbb{E}_{t-1}[C_{t,i}] &= \mathbb{E}_{t-1}[\mathbb{E}_{t-1}[C_{t,i} \mid C_{t,i-1}]] \\
&= \mathbb{E}_{t-1}[\mathbb{E}_{t-1}[\mathrm{Bin}\,(C_{t,i-1}, p_{t,2i-1}p_{t,2i}) \mid C_{t,i-1}]] \\
&= \mathbb{E}_{t-1}[C_{t,i-1} \cdot p_{t,2i-1}p_{t,2i}] \\
&= \mathbb{E}_{t-1}[C_{t,i-1}] \cdot p_{t,2i-1}p_{t,2i}.
\end{aligned}$$

By (6) and (8), we also get

$$\begin{aligned}
\mathbb{E}_{t-1}[D_{t,i}] &= \mathbb{E}_{t-1}[\mathbb{E}_{t-1}[D_{t,i} \mid C_{t,i-1}, C_{t,i}]] \\
&= \mathbb{E}_{t-1}\left[(C_{t,i-1} - C_{t,i})\frac{(1 - p_{t,2i-1})(1 - p_{t,2i})}{1 - p_{t,2i-1}p_{t,2i}}\right] \\
&= \mathbb{E}_{t-1}[C_{t,i-1}](1 - p_{t,2i-1}p_{t,2i})\frac{(1 - p_{t,2i-1})(1 - p_{t,2i})}{1 - p_{t,2i-1}p_{t,2i}} \\
&= \mathbb{E}_{t-1}[C_{t,i-1}] \cdot (1 - p_{t,2i-1})(1 - p_{t,2i}),
\end{aligned}$$

and similarly by (7), (8) and (9), we finally obtain

$$\mathbb{E}_{t-1}[E_{t,i}] = \mathbb{E}_{t-1}[C_{t,i-1}] \cdot p_{t,2i-1}(1 - p_{t,2i}). \qquad \square$$

4.2 In the initial population

An initial observation is that the all-ones bitstring cannot be sampled in the initial population P_0 with high probability since the probability of sampling

it from the uniform distribution is 2^{-n}, then by the union bound [41] it appears in the initial population of λ individuals with probability at most $\lambda \cdot 2^{-n} = 2^{-\Omega(n)}$ since we only consider the offspring population of size at most polynomial in the problem instance size n. The following lemma states the expectations of the random variables Z_t^* and Z_t (defined in (3) and (4), respectively) in the iteration $t = 0$.

Lemma 5. $\mathbb{E}[Z_0^*] = \mathcal{O}(\log \lambda)$, *and* $\mathbb{E}[Z_0] = \mathcal{O}(\log(\lambda - \mu))$.

Proof. Recall that $Z_0^* = \max\{i : C_{0,i} > 0\}$ and the definition of the function $\phi(x)$ in (1). The probability of sampling an individual with k leading 11s (where $k < m$) is $\Pr(\phi(x) = k) = (1/4)^k(1 - 1/4) = 3 \cdot 4^{-(k+1)}$, then $\Pr(\phi(x) \leq k) = 1 - 4^{-(k+1)}$. The event $\{Z_0^* \leq k\}$ implies that the λ individuals all have at most k leading 11s, i.e.,

$$\Pr(Z_0^* \leq k) = \prod_{i=1}^{\lambda} \Pr(\phi(x_0^{(i)}) \leq k) = (1 - 4^{-(k+1)})^{\lambda},$$

and $\Pr(Z_0^* > k) = 1 - (1 - 4^{-(k+1)})^{\lambda}$. Since the random variable Z_0^* is integer-valued and by $\sum_{i=1}^{k}(1/i) < 1 + \ln k$, we get

$$\begin{aligned}
\mathbb{E}[Z_0^*] &< \sum_{k=0}^{\infty} \Pr(Z_0^* > k) \\
&= \sum_{k=0}^{\infty} (1 - (1 - 4^{-(k+1)})^{\lambda}) \\
&< 1 + \int_0^{\infty} (1 - (1 - e^{-x \ln 4})^{\lambda}) \mathrm{d}x \\
&= \frac{1}{\ln 4} \int_0^1 \frac{1 - u^{\lambda}}{1 - u} du \qquad \text{(by following [20])} \\
&= \frac{1}{\ln 4} \int_0^1 \sum_{i=0}^{\lambda-1} u^i du = \frac{1}{\ln 4} \sum_{i=1}^{\lambda} \frac{1}{i} < \frac{1 + \ln \lambda}{\ln 4} = \mathcal{O}(\log \lambda),
\end{aligned}$$

which proves the first claim.

For the second claim, we take an alternative view that the random variable Z_0 denotes the number of leading 11s of the fittest individual(s) in a smaller population of the $\lambda - \mu + 1$ remaining individuals (i.e., all but the $\mu - 1$ fittest individuals in the initial population), sampled from a uniform distribution. The same line of arguments above immediately yields

$$\mathbb{E}[Z_0] < \frac{1 + \ln(\lambda - \mu + 1)}{\ln 4} = \mathcal{O}(\log(\lambda - \mu)). \qquad \square$$

4.3 In an arbitrary iteration t

By the definition of the random variable Z_t, the first $2Z_t$ marginals are set to the upper border $1 - 1/n$ in iteration $t \in \mathbb{N}$. Recall that the random variable $X_{t,i}$ denotes the number of 1s in bit position $i \in [n]$ among the μ fittest individuals, which is used to update the probabilistic model of the UMDA. We also define another random variable $Y_{t,j}$ to be the number of 11s sampled in block position $j \in [m]$, also among the μ fittest individuals in an iteration $t \in \mathbb{N}$.

Lemma 6. *For any $t \in \mathbb{N}$ that*

(a) $Y_{t,j} \sim Bin(\mu, p_{t,2j-1}p_{t,2j})$ for all $j \geq Z_t + 2$, and

(b) $X_{t,i} \sim Bin(\mu, p_{t,i})$ for all $i \geq 2Z_t + 3$.

Proof. By the definition of the random variable Z_t, we know that $C_{t,Z_t} \geq \mu$ and $C_{t,Z_t+1} < \mu$. Consider the block $j \coloneqq Z_t + 2$. We then obtain from (5) that $C_{t,j} \sim \mathrm{Bin}(C_{t,j-1}, p_{t,2j-1}p_{t,2j})$. For the $\mu - C_{t,j-1} > 0$ remaining individuals (among the μ fittest individuals), these individuals have the block $j-1$ set to $\{00, 10, 01\}$. This means that the overall fitness (or the fitness ranking) of these individuals have been already decided by the first $j-1$ blocks, and what is sampled in the block j will not have any impact on the ranking of these individuals. Therefore, there is no bias in block j among these individuals, which literally means that the number of 11s sampled here follows a binomial distribution with $\mu - C_{t,j-1}$ trials and success probability $p_{t,2j-1}p_{t,2j}$, i.e., $\mathrm{Bin}(\mu - C_{t,j-1}, p_{t,2j-1}p_{t,2j})$. Putting things together, the total number of 11s sampled in the block j among the μ fittest individuals equals

$$\begin{aligned} Y_{t,j} &\sim C_{t,j} + \mathrm{Bin}(\mu - C_{t,j-1}, p_{t,2j-1}p_{t,2j}) \\ &\sim \mathrm{Bin}(C_{t,j-1}, p_{t,2j-1}p_{t,2j}) + \mathrm{Bin}(\mu - C_{t,j-1}, p_{t,2j-1}p_{t,2j}) \\ &\sim \mathrm{Bin}(\mu, p_{t,2j-1}p_{t,2j}). \end{aligned}$$

We note that should this result hold for any block $j \geq Z_t + 2$, which proves the first statement.

For the second statement, we consider a bit position $i = 2j - 1$ in block $j = Z_t + 2$. We note that the number of 1s sampled in bit position i can be written as the sum of three components:

(1) the number of individuals with at least j leading 11s (i.e., $C_{t,j}$),

(2) the number of individuals with $j-1$ leading 11s, followed by a 10 block in block j (i.e., $E_{t,j}$), and

(3) the number of 1s sampled in bit position i among all the μ fittest individuals except the top $C_{t,j-1}$ individuals. There is no bias among these individuals, so the number of 1s here is binomially distributed with parameters $\mu - C_{t,j-1}$ and $p_{t,i}$.

We note further that the sum of $C_{t,j} + E_{t,j}$ equals the number of 1s sampled in bit position i among the $C_{t,j-1}$ fittest individuals. Thus, we get:

$$\begin{aligned} X_{t,i} &\sim C_{t,j} + E_{t,j} + \mathrm{Bin}\left(\mu - C_{t,j-1}, p_{t,i}\right) \\ &\sim \mathrm{Bin}\left(C_{t,j-1}, p_{t,i}\right) + \mathrm{Bin}\left(\mu - C_{t,j-1}, p_{t,i}\right) \\ &\sim \mathrm{Bin}\left(\mu, p_{t,i}\right). \end{aligned}$$

By the same line of argumentation, we can show that the number of 1s sampled in bit position $i+1$ is $X_{t,i+1} \sim \mathrm{Bin}\left(\mu, p_{t,i+1}\right)$, and similarly for other bit positions from $i+3$ to n. □

We now consider the block $i = Z_t + 1$, where $C_{t,i} < \mu$. The following lemma shows that if the value of the random variable $C_{t,i}$ is below a threshold, then in iteration $t+1$ the number of individuals with at least i leading 11s sampled decreases, while the number of individuals with exactly $i-1$ leading 11s followed by a 00 increases in expectation. We consider two different regimes of the selective pressure, i.e., $\gamma^* < 1/2e$ and $\gamma^* \geq 1/2e$.

Lemma 7. *Consider the block $i = Z_t + 1$ in an arbitrary iteration $t \in \mathbb{N}$, and assume further that $C_{t,i} + D_{t,i} \geq \mu$.*

(A) Let $\gamma^* = \mu/\lambda < 1/2e$. If there exists a constant $\varepsilon \in (0,1)$ such that $C_{t,i} \leq (\mu^2/\lambda)(1-\varepsilon)$, then

A.1) $\mathbb{E}_t[C_{t+1,i}] < C_{t,i}(1-\varepsilon)$,

A.2) $\mathbb{E}_t[C_{t+1,i} + D_{t+1,i}] > \mu(1+\varepsilon^2)$,

A.3) $\Pr(C_{t+1,i} + D_{t+1,i} \leq \mu) \leq e^{-\Omega(\mu)}$, *and*

A.4) $\Pr(C_{t+1,i} \geq (\mu^2/\lambda)(1-\varepsilon)) \leq e^{-\Omega(\mu^2/\lambda)}$.

(B) Let $\gamma^* = \mu/\lambda \geq 1/2e$. If there exists a constant $\varepsilon \in (0,1)$ such that $C_{t,i} \leq (\mu/2e)(1-\sqrt{\alpha})$, where $\alpha \coloneqq 2e(1+\varepsilon)(\mu/\lambda) - 1 \geq \varepsilon$, then

B.1) $\mathbb{E}_t[C_{t+1,i}] < C_{t,i}(1-\sqrt{\varepsilon})$,

B.2) $\mathbb{E}_t[C_{t+1,i} + D_{t+1,i}] > \mu(1+\varepsilon)$,

B.3) $\Pr(C_{t+1,i} + D_{t+1,i} \leq \mu) \leq e^{-\Omega(\mu)}$, *and*

B.4) $\Pr(C_{t+1,i} \geq \mu(1-\sqrt{\alpha})/2e) \leq e^{-\Omega(\lambda)}$.

Proof. The assumption implies that the two marginals in the block i will be set to $C_{t,i}/\mu$ when updating the model in iteration t. Statement (A.1) is trivial since

$$\begin{aligned}
\mathbb{E}_t[C_{t+1,i}] &= \mathbb{E}_t[C_{t+1,i-1}] \cdot p_{t+1,2i-1}p_{t+1,2i} \\
&= \lambda(1-1/n)^{2(i-1)}(C_{t,i}/\mu)(C_{t,i}/\mu) \\
&< \lambda(C_{t,i}/\mu)(\mu/\lambda)(1-\varepsilon) \\
&= C_{t,i}(1-\varepsilon).
\end{aligned}$$

Noting also that $C_{t,i}/\mu \leq (\mu/\lambda)(1-\varepsilon) < (1-\varepsilon)/2e < (1-\varepsilon)/2$. The statement (A.2) can be shown as follows.

$$\begin{aligned}
\mathbb{E}_t[C_{t+1,i} + D_{t+1,i}] &= \lambda(1-1/n)^{2(i-1)}((C_{t,i}/\mu)^2 + (1-C_{t,i}/\mu)^2) \\
&\geq (\lambda/e)(1-2(C_{t,i}/\mu)(1-C_{t,i}/\mu)) \\
&\geq \lambda(1-(1-\varepsilon)(1-(1-\varepsilon)/2)) \\
&= (\lambda/2e)(1+\varepsilon^2) \\
&> \mu(1+\varepsilon^2).
\end{aligned}$$

For the statement (A.3), we now associate each of the λ individuals in the population with an indicator random variable , which is set to 1 if the individual has $i-1$ leading 11s, followed by either a 11 or a 00. There are λ such indicators, and we are interested in their sum, which is identical to the sum of $D_{t+1,i} + C_{t+1,i}$. By statement (A.2), the expectation of the sum is at least $\mu(1+\varepsilon^2) = \mu/(1-\delta)$ for some constants $\varepsilon \in (0,1)$ and $\delta \coloneqq 1 - 1/(1+\varepsilon^2)$. Then, by a Chernoff bound (see Lemma 1 in the Appendix) the probability that the sum is at most $(1-\delta)\cdot\mu/(1-\delta) = \mu$ is at most $e^{-(\delta^2/2)\cdot\mu/(1-\delta)} = e^{-\Omega(\mu)}$.

For the statement (A.4), we note that $C_{t+1,i}$ is stochastically dominated by another random variable $\tilde{C}$, which is binomially distributed with λ trials and success probability $(\mu/\lambda)^2(1-\varepsilon)^2$. Note that $\mathbb{E}[\tilde{C}] = (\mu^2/\lambda)(1-\varepsilon)^2 = \Omega(\mu^2/\lambda)$; thus, we can rewrite $(\mu^2/\lambda)(1-\varepsilon) = \mathbb{E}[\tilde{C}]/(1-\varepsilon) = (1+\varepsilon')\mathbb{E}[\tilde{C}]$ for some other constant $\varepsilon' \coloneqq 1/(1-\varepsilon) - 1 > 0$. By a Chernoff bound, we then obtain

$$\begin{aligned}
\Pr(C_{t+1,i} \geq (\mu^2/\lambda)(1-\varepsilon)) &\leq \Pr(\tilde{C} \geq (\mu^2/\lambda)(1-\varepsilon)) \\
&= \Pr(\tilde{C} \geq (1+\varepsilon')\mathbb{E}[\tilde{C}]) \\
&\leq e^{-(\varepsilon')^2 \cdot \mathbb{E}[\tilde{C}]/3} \\
&= e^{-\Omega(\mu^2/\lambda)},
\end{aligned}$$

which completes proof of statement (A.4).

To prove statement (B.1), by (1) and (2), we get

$$\begin{aligned}
C_{t,i}/\mu &\leq (1/2e)(1 - \sqrt{2e(1+\varepsilon)(\mu/\lambda) - 1}) \\
&\leq (1/2e)(1 - \sqrt{2e(1+\varepsilon)(1/2e) - 1}) \\
&= (1 - \sqrt{\varepsilon})/(2e) \\
&< (1 - \sqrt{\varepsilon})/2.
\end{aligned}$$

Therefore, statement (B.1) can be shown as follows.

$$\begin{aligned}
\mathbb{E}_t[C_{t+1,i}] &< \lambda(C_{t,i}/\mu)^2 \\
&= (\lambda/\mu)(C_{t,i}/\mu)C_{t,i} \\
&\leq (2e)((1 - \sqrt{\varepsilon})/2e)C_{t,i} \\
&= C_{t,i}(1 - \sqrt{\varepsilon}).
\end{aligned}$$

For statement (B.2), we note that $C_{t,i}/\mu \leq (1 - \sqrt{\alpha})/2e < (1 - \sqrt{\alpha})/2$ and then obtain

$$\begin{aligned}
\mathbb{E}_t[C_{t+1,i} + D_{t+1,i}] &\geq (\lambda/e)(1 - 2(C_{t,i}/\mu)(1 - C_{t,i}/\mu)) \\
&> (\lambda/e)(1 - (1 - \sqrt{\alpha})(1 - (1 - \sqrt{\alpha})/2))) \\
&= (\lambda/e)(1 - (1 - \sqrt{\alpha}) + (1/2)(1 - \sqrt{\alpha})^2) \\
&= (\lambda/e)(\sqrt{\alpha} + (1/2)(1 - 2\sqrt{\alpha} + \alpha)) \\
&= (\lambda/2e)(1 + \alpha) \\
&= (\lambda/2e)2e(1+\varepsilon)(\mu/\lambda) \\
&= \mu(1+\varepsilon).
\end{aligned}$$

The statement (B.3) follows similarly to the proof of statement (A.3). For the statement (B.4), we employ a similar approach used in (A.4), where we choose $\tilde{C} \sim \text{Bin}\left(\lambda, ((1-\sqrt{\alpha})/2e)^2\right)$ and $\mathbb{E}[\tilde{C}] = \lambda((1-\sqrt{\alpha})/2e)^2 = \Omega(\lambda)$. We also note that

$$\begin{aligned}
\mu(1-\sqrt{\alpha})/2e &= \mathbb{E}[\tilde{C}]/((\lambda/\mu)((1-\sqrt{\alpha})/2e)) \\
&\geq \mathbb{E}[\tilde{C}]/(1-\sqrt{\alpha}) \\
&\geq \mathbb{E}[\tilde{C}]/(1-\varepsilon) \\
&= (1+\varepsilon')\mathbb{E}[\tilde{C}]
\end{aligned}$$

for some other constant $\varepsilon' = 1/(1-\varepsilon) - 1 > 0$ and $\alpha \geq \varepsilon$. Since $\tilde{C}$ stochastically dominates $C_{t+1,i}$, we get by a Chernoff bound that

$$\begin{aligned}
\Pr(C_{t+1,i} \geq \mu(1-\sqrt{\alpha})/2e) &\leq \Pr(\tilde{C} \geq \mu(1-\sqrt{\alpha})/2e) \\
&\leq \Pr(\tilde{C} \geq (1+\varepsilon')\mathbb{E}[\tilde{C}]) \\
&\leq e^{-(\varepsilon')^2 \cdot \mathbb{E}[\tilde{C}]/3} \\
&\leq e^{-\Omega(\lambda)},
\end{aligned}$$

which completes the proof. □

We note that combining the two conditions of the statement (A) in Lemma 7 yields $C_{t,i} < \mu(1-\varepsilon)/2e$ for some small constant $\varepsilon \in (0,1)$, and the same calculation for the statement (B) yields $C_{t,i} < \mu(1-\sqrt{\varepsilon})/2e$. We observe that the two upper bounds are asymptotically identical, and since the statement (A) considers the case of high selective pressure, we will in the remainder of the paper form our arguments based on this result only. Previous studies [7, 38] show that the UMDA only works under a sufficiently high selective pressure (on the LeadingOnes function). In other words, if we can show that the UMDA cannot optimise the DLB function efficiently for some selective pressure $\gamma^* < 1/2e$, this result will highly likely hold for any selective pressure $\gamma^* \geq 1/2e$.

Furthermore, Lemma 7 also tells us that in an iteration $t \in \mathbb{N}$ if the block $i = Z_t + 1$ consists of 00s and 11s only among the μ fittest individuals, and the number of 11s is below a threshold $C_{t,i} \leq (\mu^2/\lambda)(1-\varepsilon)$ for some small constant $\varepsilon \in (0,1)$, then in expectation the number of 11s sampled in the next iteration will shrink, while that of 00s will expand, and the block i still

consists of 00s and 11s only among the μ fittest individuals. Mathematically speaking, we obtain

$$\mathbb{E}_t[C_{t,i} - C_{t+1,i}] > \varepsilon C_{t,i}.$$

This means that there is a (multiplicative) drift towards the value of zero on the stochastic process $(C_{t,i})_{t\in\mathbb{N}}$. By the multiplicative drift theorem [16], the random variable $C_{t,i}$ will hit the value of zero in an $\mathcal{O}(\log \mu)$ expected time. Once this has happened, we will show that the UMDA requires at least $e^{\Omega(\mu)}$ iterations in expectation to sample at least

$$\theta \coloneqq (\mu^2/\lambda)(1-\varepsilon) = \gamma^*\mu(1-\varepsilon) \tag{11}$$

11s to gain enough momentum to escape the 'trap' in the block $Z_t + 1$ (another way of saying this is to repair the specified block).

Furthermore, we note so far that Lemma 7 assumes that there are only 11s and 00s among the μ fittest individuals in block $i = Z_t + 1$, which make the two corresponding marginals simultaneously set to $C_{t,i}/\mu$. This is, however, not strictly necessary because the UMDA updates each marginal using the total number of 1-bits sampled in the bit position, so as long as the number of 1s in each bit position is still below the threshold θ, then all results in Lemma 7 still hold.

Recall that we aim at showing an $e^{\Omega(\mu)}$ lower bound on the runtime of the UMDA on the DLB function. This lower bound will be obtained if we can show that there exists a block $i = Z_t + 1 < m$ between Z_0 and $m = n/2$, where the two marginals are deceived to reach the lower bound $1/n$, and then the UMDA has to wait a long time (in terms of iterations) while the block is being repaired.

Lemma 8. *Let $c \log n \leq \mu = o(n)$ for some sufficiently large constant $c > 0$. If there exists a constant $k < m$ such that $Z_t \leq k - 2$ for any time $t \in \mathbb{N}$, then for any $j \in [2k-1, n]$ that*

(a) $\mathbb{E}[p_{t,j}] = 1/2$,

(b) $\mathbb{E}[X_{t,j}] = \mu/2$, *and*

(c) $Var[X_{t,j}] \geq (\mu^2/4)(1-o(1))(1-(1-1/\mu)^t)$.

Proof. For readability, we omit the index j through out the proof. Recall that $p_t = \max\{1/n, \min\{1-1/n, X_{t-1}/\mu\}\}$. By the definition of expectation, we get

$$\begin{aligned}\mathbb{E}[p_t] = (1/n) \cdot \Pr\left(X_{t-1} = 0\right) + (1 - 1/n) \cdot \Pr\left(X_{t-1}\mu\right) \\ + \sum_{k=1}^{\mu-1} (k/\mu) \cdot \Pr\left(X_{t-1} = k\right).\end{aligned} \tag{12}$$

We note further that

$$\mathbb{E}[X_{t-1}] = \sum_{k=0}^{\mu} k \Pr(X_{t-1} = k) = \mu \Pr(X_{t-1} = \mu) + \sum_{k=1}^{\mu-1} k \Pr(X_{t-1} = k),$$

from which we then obtain

$$\sum_{k=1}^{\mu-1} k \cdot \Pr(X_{t-1} = k) = \mathbb{E}[X_{t-1}] - \mu \cdot \Pr(X_{t-1} = \mu). \tag{13}$$

Substituting (13) into (12) yields

$$\mathbb{E}[p_t] = (1/\mu)\mathbb{E}\left[X_{t-1}\right] + (1/n)\left(\Pr\left(X_{t-1} = 0\right) - \Pr\left(X_{t-1} = \mu\right)\right). \tag{14}$$

We note by Lemma 6 that X_t follows a binomial distribution with μ trials and success probability $p_{t,j}$, which means that there is no bias towards any border in the stochastic process $(X_t)_{t\in\mathbb{N}}$. Due to this symmetry, we get

$$\Pr\left(X_{t-1} = \mu\right) = \Pr\left(X_{t-1} = 0\right). \tag{15}$$

Furthermore, by the tower rule we also have

$$\mathbb{E}[X_{t-1}] = \mathbb{E}[\mathbb{E}[X_{t-1} \mid p_{t-1}]] = \mathbb{E}\left[\mathbb{E}\left[\mathrm{Bin}\left(\mu, p_{t-1}\right) \mid p_{t-1}\right]\right] = \mu \cdot \mathbb{E}[p_{t-1}] \tag{16}$$

Substituting (15) and (16) into (14) yields $\mathbb{E}[p_t] = \mathbb{E}[p_{t-1}]$. Then by induction on time, we obtain

$$\mathbb{E}[p_t] = \mathbb{E}[p_{t-1}] = \mathbb{E}[p_{t-2}] = \ldots = \mathbb{E}[p_0] = 1/2,$$

which completes the proof of statement (a).

Statement (b) follows from (16) that

$$\mathbb{E}[X_t] = \mu \cdot \mathbb{E}[p_t] = \mu/2.$$

For statement (c), we note by the tower rule and Lemma 3 that

$$\begin{aligned}\mathbb{E}[X_t^2] &= \mathbb{E}[\mathbb{E}[X_t^2 \mid p_t]] = \mathbb{E}[\mu p_t(p_t(\mu-1)+1)] \\ &= \mu(\mu-1)\mathbb{E}[p_t^2] + \mu\mathbb{E}[p_t] = \mu(\mu-1)\mathbb{E}[p_t^2] + \mu/2.\end{aligned} \tag{17}$$

By the definition of expectation, we also have

$$\begin{aligned}\mathbb{E}[p_t^2] &= (1/n)^2 \cdot \Pr(X_{t-1}=0) + (1-1/n)^2 \cdot \Pr(X_{t-1}=\mu) \\ &\quad + \sum_{k=1}^{\mu-1}(k/\mu)^2 \cdot \Pr(X_{t-1}=k),\end{aligned}$$

which by noting that

$$\mathbb{E}[X_{t-1}^2] = \mu^2 \cdot \Pr(X_{t-1}=\mu) + \sum_{k=1}^{\mu-1} k^2 \cdot \Pr(X_{t-1}=k)$$

satisfies

$$\begin{aligned}\mathbb{E}[p_t^2] &= (1/n)^2 \cdot \Pr(X_{t-1}=0) + (1-1/n)^2 \cdot \Pr(X_{t-1}=\mu) \\ &\quad + (1/\mu^2)(\mathbb{E}[X_{t-1}^2] - \mu^2 \Pr(X_{t-1}=\mu)) \\ &= (1/\mu^2)\mathbb{E}[X_{t-1}^2] + (1/n)^2 \cdot \Pr(X_{t-1}=0) \\ &\quad + ((1-1/n)^2 - 1) \cdot \Pr(X_{t-1}=\mu) \\ &= (1/\mu^2) \cdot \mathbb{E}[X_{t-1}^2] - (2/n)(1-1/n) \cdot \Pr(X_{t-1}=\mu)\end{aligned}$$

By (15), we can simplify the expression above further as follows.

$$\begin{aligned}\mathbb{E}[p_t^2] &= (1/\mu^2)\mathbb{E}[X_{t-1}^2] - (2/n)(1-1/n) \cdot \Pr(X_{t-1}=\mu) \\ &\geq (1/\mu^2)\mathbb{E}[X_{t-1}^2] - (2/n)(1-1/n)\end{aligned} \tag{18}$$

since $\Pr(X_{t-1}=\mu) \leq 1$. Substituting (18) into (17) yields

$$\begin{aligned}\mathbb{E}[X_t^2] &\geq (1-1/\mu)\mathbb{E}[X_{t-1}^2] - 2(\mu/n)(\mu-1)(1-1/n) + \mu/2 \\ &= (1-1/\mu)\mathbb{E}[X_{t-1}^2] + (\mu/2)(1-o(1))\end{aligned}$$

since $\mu = o(n)$. We now obtain a recurrence relation for the expectation of X_t^2 w.r.t. time t and by $\sum_{i=1}^{n} c^i = (c^{n+1}-1)/(c-1)$ for any $c \neq 1$, we then get

$$\begin{aligned}\mathbb{E}[X_t^2] &\geq (1-1/\mu)^t\mathbb{E}[X_0^2] + (\mu/2)(1-o(1))\sum_{i=0}^{t-1}(1-1/\mu)^i \\ &= (1-1/\mu)^t\mathbb{E}[X_0^2] + (\mu/2)(1-o(1)) \cdot \frac{(1-1/\mu)^t - 1}{(1-1/\mu)-1} \\ &= (1-1/\mu)^t\mathbb{E}[X_0^2] + (\mu^2/2)(1-o(1))(1-(1-1/\mu)^t)\end{aligned}$$

which by $\mathbb{E}[X_0^2] = \mu(1/2)((1/2)(\mu-1)+1) = \mu(\mu+1)/4$ (see Lemma 3 for $X_0 \sim \mathrm{Bin}(\mu, 1/2)$) satisfies

$$\begin{aligned}
&\geq (1-1/\mu)^t \mu(\mu+1)/4 + (\mu^2/2)(1-o(1))(1-(1-1/\mu)^t) \\
&= (\mu^2/2)(1-o(1)) - (1-1/\mu)^t((\mu^2/2)(1-o(1)) - \mu(\mu+1)/4) \\
&= (\mu^2/2)(1-o(1)) - (1-1/\mu)^t(\mu^2/4)(1-o(1))
\end{aligned}$$

Thus, we obtain

$$\begin{aligned}
\mathrm{Var}[X_t] &= \mathbb{E}[X_t^2] - \mathbb{E}[X_t]^2 \\
&\geq (\mu^2/2)(1-o(1)) - (1-1/\mu)^t(\mu^2/4)(1-o(1)) - (\mu/2)^2 \\
&\geq (\mu^2/4)(1-o(1)) - (1-1/\mu)^t(\mu^2/4)(1-o(1)) \\
&\geq (\mu^2/4)(1-o(1))(1-(1-1/\mu)^t),
\end{aligned}$$

which completes the proof of statement (c). □

One should not confuse the result implied by the statement (a) in Lemma 8 with the actual value of the marginals in an arbitrary iteration $t \in \mathbb{N}$. For the UMDA without borders, Friedrich et al. [22] showed that even when the expectation stays at $1/2$, the actual value of the marginal in iteration t can be close to the trivial lower or upper border due to its large variance. Recently, Zheng et al. [54] showed that this actually happens within an $\Theta(\mu)$ expected number of iterations.

Furthermore, in case of no borders, the variance of $X_{t,j}$ for any bit $j \in [2k-1, n]$, where k is defined in Lemma 8, can be expressed as a function of time as $\mathrm{Var}^*[X_{t,j}] = (1-(1-1/\mu)^t)(\mu^2/4)$, which can be derived by applying the law of total variance on the martingale $X_{t,j} \sim \mathrm{Bin}(\mu, X_{t-1,j}/\mu)$ (see [22, Lemma 7, Corrolary 9] for a similar derivation for the cGA without borders). Surprisingly, the statement (c) in Lemma 8 tells us that the variance of $X_{t,j}$ for the UMDA with borders grows asymptotically in the same order as the variance for the UMDA without borders. The following lemma shows that after a sufficiently long time the probability that one will find the value of the random variable $X_{t,j}$ smaller than the threshold θ or greater than $\mu - \theta$, where θ is defined in (11), with a constant probability.

Lemma 9. *For any $c \in \mathbb{R}$ and even number $\mu \in \mathbb{N}$, let $X \in \{0, \ldots, \mu\}$ be a random variable with $\mathrm{Var}[X] \geq c\mu^2$ and*

$$\Pr(X = \mu/2 - i) = \Pr(X = \mu/2 + i)$$

for all $i \in [\mu/2]$. *Then,*

$$\Pr\left(X \leq \theta\right) \geq \frac{2(c - (1/2 - \gamma^*)^2)}{1 - 4(1/2 - \gamma^*)^2}.$$

Proof. Due to symmetry, we have $\mathbb{E}\left[X\right] = \mu/2$ and

$$p := \Pr\left(X \leq \theta\right) = \Pr\left(X \geq \mu - \theta\right).$$

The lower bound on the variance of X implies

$$\begin{aligned}
c\mu^2 \leq \mathrm{Var}[X] &= \sum_{i=0}^{\mu} \Pr\left(X = i\right)\left(i - \mathbb{E}\left[X\right]\right)^2 \\
&\leq \Pr\left(X \leq \theta\right)\frac{\mu^2}{4} + \Pr\left(\theta < X < \mu - \theta\right)(\mu/2 - \theta)^2 \\
&\quad + \Pr\left(X \geq \mu - \theta\right)\frac{\mu^2}{4} \\
&= \frac{p\mu^2}{4} + (1 - 2p)(\mu/2 - \theta)^2 + \frac{p\mu^2}{4} \\
&= (\mu/2 - \theta)^2 + (\mu^2/2 - 2(\mu/2 - \theta)^2)p
\end{aligned}$$

Solving the inequality above for p gives

$$p \geq \frac{2(c\mu^2 - (\mu/2 - \theta)^2)}{\mu^2 - 4(\mu/2 - \theta^2} = \frac{2(c - (1/2 - \gamma^*)^2)}{1 - 4(1/2 - \gamma^*)^2}.$$ □

4.4 Exponential runtime in case of low selective pressure

In this section, we will show that the UMDA requires an $e^{\Omega(\mu)}$ expected runtime to optimise the DLB function. To proceed, we will consider two phases:

1) until the algorithm gets stuck at a block $Z_t + 1$, and

2) while the algorithm is getting stuck and afterwards.

Recall that the algorithm gets stuck at some value $Z_t < m$ when the number of 1s in each bit position in the block $Z_t + 1$ among the μ fittest individuals is at most $\theta = (\mu^2/\lambda)(1 - \varepsilon)$ for some small constant $\varepsilon \in (0, 1)$. We will shows that phase 1 will last for $\Omega(\mu)$ iterations. The following lemma shows that the all-ones bitstring cannot be sampled during the first $\Omega(\mu)$ iterations with high probability.

Lemma 10. *With probability* $1 - 2^{-\Omega(n)}$*, the all-ones bitstring cannot be sampled during the first* 2.92μ *iterations of the* UMDA *optimising the* DLB *function.*

Proof. The proof is inspired by [32, Lemma 9]. We will upper bound the probability of an arbitrary bit position $j \in \mathbb{N} \cap [2Z_0 + 3, n]$ exceeding $99/100$ during the first $\Omega(\mu)$ iterations. For readability, we omit the index j and consider the potential $\phi_t \coloneqq X_t^2$. By Lemma 6, we can pessimistically assume we are not at the borders, which implies that $X_{t+1} \sim \text{Bin}(\mu, X_t/\mu)$, and by Lemma 3 the expected single-step change is

$$\mathbb{E}_t[\phi_{t+1} - \phi_t] = \mu \frac{X_t}{\mu}\left(\frac{X_t}{\mu}(\mu - 1) + 1\right) - X_t^2 = X_t\left(1 - \frac{X_t}{\mu}\right) < \frac{\mu}{4}.$$

Let $T \coloneqq \min\{t \in \mathbb{N} \mid X_t \geq 99\mu/100\}$, i.e., the first hitting time of the value of $99\mu/100$ in the stochastic process $(X_t)_{t\in\mathbb{N}}$. We have a Markov chain ϕ_T with process $\phi_t = X_t^2$ starting at $(\mu/2)^2$ and then progressing by $\phi_{t+1} - \phi_t$ for T iterations. We then get

$$\mathbb{E}[\phi_T] = \left(\frac{\mu}{2}\right)^2 + \sum_{t=0}^{T-1} \mathbb{E}[\mathbb{E}_t[\phi_{t+1} - \phi_t]] < \left(\frac{\mu}{2}\right)^2 + T \cdot \frac{\mu}{4}.$$

Using Markov's inequality for $k > 1$ yields

$$\Pr\left(\phi_T \geq k\left(\frac{\mu^2}{4} + T \cdot \frac{\mu}{4}\right)\right) \leq \Pr\left(\phi_T \geq k \cdot \mathbb{E}\left[\phi_T\right]\right) \leq \frac{1}{k}.$$

We want that $(99\mu/100)^2 \geq k\left(\mu^2/4 + T \cdot \mu/4\right)$ since then

$$\Pr\left(\phi_T \geq \left(\frac{99\mu}{100}\right)^2\right) \leq \Pr\left(\phi_T \geq k\left(\frac{\mu^2}{4} + T \cdot \frac{\mu}{4}\right)\right) \leq \frac{1}{k}.$$

We get $T \leq \mu((4/k)(99/100)^2 - 1)$, which is positive as long as $k \in (1, 4 \cdot (99/100)^2)$. Thus, we can choose a value of $k = 1.00001$ and then obtain $T \leq 2.92\mu$. During the first $\Omega(\mu)$ iterations, the probability of an arbitrary marginal $j \in \mathbb{N} \cap [2Z_0 + 3, n]$ to exceed $99/100$ is at most a constant $1/k < 1$. By Lemma 5, in expectation there are at most $(1/k) \cdot (n - \mathbb{E}[Z_0]) = (1/k) \cdot (n - \mathcal{O}(\log(\lambda - \mu))) < n/k$ marginals exceeding $99/100$, and by Chernoff bound, with probability $1 - 2^{-\Omega(n)}$ there are at most $(1 + \delta)(n/k)$ such marginals for a constant $\delta > 0$ and at least $(1 - (1+\delta)/k)n = \Omega(n)$ marginals are still below $99/100$ during the first 2.92μ iterations. Thus, the probability of sampling the all-ones bitstring is upper bounded by $(99/100)^{\Omega(n)} = 2^{-\Omega(n)}$. □

We now show that the all-ones bitstring cannot be sampled with probability $1-2^{-\Omega(n)}$ while the UMDA is getting stuck at a block. To do this, we need the following two lemmas, where the first one implies the independent sampling at the $2(m-(Z_t+1))$ remaining bit positions, and the other states that the expected values of the marginals of these remaining bits will highly likely stay around the value of $1/2$.

Lemma 11. *Consider the situation of Lemma 8. Then, the events of sampling of 11s in any block from k to m are pairwise independent. Furthermore, the all-ones bitstring cannot be sampled with probability at least $1-2^{-\Omega(n)}$.*

Proof. Recall the definition of the constant k in Lemma 8. We have observed in Lemma 6 that the number of 11s sampled among the μ fittest individuals in an arbitrary block $j \geq k$ in iteration $t \in \mathbb{N}$ is binomially distributed with μ trials and success probability $p_{t,2j-1}p_{t,2j}$. This result also implies that sampling a 11 at a block j_1 is independent of sampling a 11 at a block j_2, for any $j_1, j_2 \in [k,m]$ and $j_1 \neq j_2$, which proves the first claim.

For the second claim, we prove by considering the number of 11s sampled in an offspring between blocks k and m. By Lemma 8, the probability of sampling a 11 in a block is $1/4$, so the expected number of 11s sampled between blocks k and m (i.e., there are $m-k+1=\Theta(n)$ blocks in total) is given by

$$\mathbb{E}[Y_t] = (m-k+1)\cdot(1/2)^2 = \Theta(n).$$

Then, by the Chernoff-Hoeffding bound, the probability of sampling these blocks all as 11s is at most $e^{-\Omega(n)}$. □

We are now ready to prove an exponential runtime of the UMDA on function DLB when the selective pressure is $\gamma^* = \Omega(1)$.

Theorem 6. *The expected runtime of the* UMDA *with the parent population size $c\log n \leq \mu = o(n)$ for some sufficiently large constant $c > 0$, and parent and offspring population sizes satisfying $\frac{\mu}{\lambda} > \frac{14}{1000}$ is $e^{\Omega(\mu)}$ on the* DLB *function.*

Proof. The all-ones bitstring cannot be sampled during the first $\Omega(\mu)$ iterations with probability $1-2^{-\Omega(n)}$ (by Lemma 10). After roughly 2.92μ iterations, we obtain from Lemma 8 that $\mathrm{Var}[X_t] \geq (1-o(1))(\mu^2/4)(1-1/e^{2.92})$.

If we choose the selective pressure $\gamma^* > \frac{14}{1000}$, then by Lemma 9 for $c = (1/4)(1 - 1/e^{2.92})(1 - o(1))$, we obtain a constant lower bound on the probability that X_t drops below the threshhold value θ, defined in (11). This also means that we can find with probability $\Omega(1)$ that in any bit position in the block $Z_t + 2$ there are fewer than θ 1s sampled, where θ is defined in (11). Assume in iteration $t' = t + 1$ that $Z_{t'} = Z_t + 1$, then with probability $\Omega(1)$ the UMDA will get stuck at block $Z_{t'} + 1 = Z_t + 2$ in iteration t'. Assume now that we are in iteration t', there is a multiplicative drift towards the value of zero in the stochastic process $(C_{t'+\Delta t,i})_{\Delta t \in \mathbb{N}}$ in block $i \coloneqq Z_{t'} + 1$. By multiplicative drift theorem, the number of 11s sampled there will reduce to zero within an $\mathcal{O}(\log \mu)$ expected number of iterations. In particular, we note by the statements (A.3) and (A.4) of Lemma 7 that after the number of 11s has dropped below the threshold θ, the event of sampling at least θ 11s in the next iteration in block i or sampling at least one 01 or 10 among the μ fittest individuals is at most

$$e^{-\Omega(\mu)} + e^{-\Omega(\mu^2/\lambda)} \leq e^{-\Omega(\gamma^*\mu)} = e^{-\Omega(\mu)}$$

since we consider only $\gamma^* > \frac{14}{1000}$. Thus, the UMDA requires at least $1/e^{-\Omega(\mu)} = e^{\Omega(\mu)}$ iterations in expectation until this block has been repaired. But then, the algorithm will likely get stuck in some of the following blocks. By Lemma 5, the expected number of times the algorithm gets trapped is

$$\Omega(1) \cdot (n/2 - \mathbb{E}[Z_0]) \geq \Omega(1) \cdot (n - \mathcal{O}(\log(\lambda - \mu))) = \Omega(n).$$

Therefore, the expected number of iterations of the UMDA on the DLB function is at least $\Omega(n) \cdot e^{\Omega(\mu)}$. We also note that so far in the proof we have always assumed the co-occurrence of the following two events:

(A) The all-ones bitstring will not be sampled during the first $\Omega(\mu)$ iterations with probability $1 - 2^{-\Omega(n)}$ (by Lemma 10).

(B) The all-ones bitstring cannot be sampled while the UMDA is getting stuck at a block with probability $1 - 2^{-\Omega(n)}$ (by Lemma 11).

By union bound, the success probability is still $1 - 2^{-\Omega(n)}$. By the law of total expectation [41] and noting further that the UMDA performs λ function evaluations in every iteration, the overall expected runtime is lower-bounded by

$$(1 - 2^{-\Omega(n)}) \cdot \lambda \cdot (\Omega(\mu) + \Omega(n) \cdot e^{\Omega(\mu)}) = e^{\Omega(\mu)}. \qquad \square$$

We observe that if the parent population size is $\mu = \Theta(\log n)$, Theorem 6 yields a lower bound of $n^{\Omega(1)}$ for any selective pressure $\gamma^* > \frac{14}{1000}$. The intuition is that when the population size μ is small. the threshold θ is not too large, meaning that the UMDA only needs to sample a few 1-bits in order to escape the 'current trap' in an arbitrary iteration. Larger population sizes, such as $\mu = \Omega(n^\varepsilon)$ for some constant $\varepsilon > 0$, will result in an exponential lower bound of $2^{\Omega(n^\varepsilon)}$ on the expected runtime of the UMDA on the DLB function.

4.5 Extremely high selective pressure may help

We now apply Theorem 1 to derive an $\mathcal{O}\left(n^3\right)$ expected runtime for the UMDA on the DLB function under extremely high selective pressures of $\gamma^* = \mathcal{O}\left(1/\mu\right)$ (or $\lambda = \Omega(\mu^2)$).

Theorem 7. *The* UMDA *with the parent population size $\mu \geq c \log n$ for a sufficiently large constant $c > 0$, and the offspring population size $\lambda \geq (1+\delta)e\mu^2$ for any constant $\delta > 0$, has expected optimisation time $\mathcal{O}(n\lambda \log \lambda + n^3)$ on the* DLB *function.*

Proof. Recall that levels are defined as in (2). There are $m \coloneqq (n/2) + 1$ levels from A_0 to A_m.

For condition (G2), for any level $j \in \{0\} \cup [m-2]$ satisfying $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda = \mu$ and $|P_t \cap A_{\geq j+1}| \geq \lambda\gamma$ for some $\gamma \in (0, \gamma_0]$, we seek a lower bound $(1+\delta)\gamma$ for $\Pr\left(y \in A_{\geq j+1}\right)$ where y is sampled from the model p_{t+1}. The given conditions on j imply that the μ fittest individuals of P_t have at least j leading 11s and among them at least $\lceil \gamma\lambda \rceil$ have at least $j+1$ leading 11s. Hence, $p_{t+1,i} = 1 - 1/n$ for $i \in [2j]$, and for $i \in \{2j+1, 2j+2\}$ that $p_{t+1,i} \geq \max(\min(1-1/n, \gamma\lambda/\mu), 1/n) \geq \min(1-1/n, \gamma/\gamma_0)$, so

$$\begin{aligned}\Pr\left(y \in A_{\geq j+1}\right) &\geq \prod_{i=1}^{j+1} p_{t+1,i} \geq \left(1 - \frac{1}{n}\right)^{2j} \left(\frac{\gamma}{\gamma_0}\right)^2 \\ &\geq \frac{1}{e}\left(\frac{\gamma}{\gamma_0}\right)^2 = \frac{\lambda\gamma}{e\mu^2} \geq (1+\delta)\gamma\end{aligned}$$

due to $\gamma_0 \geq \gamma \geq 1/\lambda$, and $\lambda \geq (1+\delta)e\mu^2$ for any constant $\delta > 0$. Therefore, condition (G2) is now satisfied.

For condition (G1), for any level $j \in \mathbb{N} \cap [0, m-1]$ satisfying $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda = \mu$ we seek a lower bound z_j on $\Pr(y \in A_{\geq j+1})$. Again the condition on level j implies that $p_{t+1,i} = 1 - 1/n$ for $i \in [2j]$. Due to the imposed lower margin, we can assume pessimistically that $p_{t+1,2j+1} = p_{t+1,2j+2} \geq 1/n$. Hence,

$$\Pr(y \in A_{\geq j+1}) \geq \left(1 - \frac{1}{n}\right)^{2j} \left(\frac{1}{n}\right)^2 \geq \frac{1}{en^2} =: z_j.$$

So, (G1) is satisfied for $z_j \coloneqq 1/(en^2)$.

Considering (G3), because δ is a constant, and both $1/z_*$ and m are $\mathcal{O}(n)$, there must exist a constant $c > 0$ such that $\mu \geq c \log n \geq (4/\delta^2) \ln(128m/(z_* \delta^2))$. Note that $\lambda = \mu/\gamma_0$, so (G3) is satisfied.

All conditions of Theorem 1 are satisfied, so the expected optimisation time of the UMDA on the DLB function is

$$\mathcal{O}\left(\sum_{j=1}^{n/2} \left(\lambda \ln \lambda + n^2\right)\right) = \mathcal{O}\left(n\lambda \log \lambda + n^3\right). \qquad \square$$

One might say that the failure of the UMDA to optimise the DLB problem is not necessarily coming from the pairwise deception, but from the low selective pressure in the same way that it struggles on the LeadingOnes function [38]. However, as a final remark, we think that this claim is inaccurate. In fact, for the range of selective pressures considered, it has been shown that the UMDA optimises other non-deceptive problems easily in polynomial expected runtime, including LeadingOnes [7, 38] and OneMax [7].

Non-elitist algorithms using (μ, λ)-selection are typically efficient when $\mu/\lambda < 1/e$, i.e., below the error threshold (see [34]). In contrast, to show that the UMDA optimises DLB efficiently, we need to decrease this ratio significantly so that we get $\mu/\lambda = \mathcal{O}(1/\mu)$, i.e., extremely high selective pressure. We do not normally see such high selective pressures in practical applications of the UMDA. Furthermore, under an $\mathcal{O}(1/\mu)$ selective pressure the UMDA selects few fittest individuals to update the model. In this extreme situation, we believe that the UMDA degenerates into the $(1, \lambda)$ EA (see Figure 1 for some empirical result of the UMDA with extreme selective pressure and simple EAs), where only the fittest individual(s) are selected to the next generation.

We have already shown that simple EAs can optimise the DLB function easily under normal selective pressure. For this reason, we think that comparing the UMDA and simple EAs in this (extreme) regime of the selective pressure is not very interesting.

5 Experiments

In the previous sections, we showed that the UMDA fails to optimise DLB since it does not remember what it has learned so far during the optimisation process. This is due to the lack of ability to capture interactions between bits.

To complement the theoretical investigation, we studied empirically the behaviour of the UMDA in the case of normal selective pressure, and in the case of extremely high selective pressure. As a base-line, we also considered the standard (μ, λ) EA. Figure 1 shows the results of 100 repetitions of the UMDA and the (μ, λ) EA on the DLB problem of size $n = 200$. The boxplots show the interquartile range of the number of correct blocks obtained by the fittest individual in the population, as a function of the number of fitness evaluations t. The maximal number of correct blocks is $n/2 = 100$. Each run was terminated after 10^6 function evaluations. The UMDA in the extreme selective pressure setting uses the parameters $\mu = 10$ and $\lambda = 1000$. The boxplots show that the algorithm in this setting converges to a non-optimal equilibirum position (around 20-30 correct blocks), as predicted by the theoretical analysis. The UMDA in the normal selective pressure setting uses the parameters $\mu = 200$ and $\lambda = 1000$. The boxplots show that the algorithm in this setting improves steadily, again as predicted by the theoretical analysis. The (μ, λ) EA uses the parameters $\mu = 200, \lambda = 1000$, and bitwise mutation rate $1/n$. The boxplots show that the algorithm has a steady progress, as predicted by the theoretical analysis.

We now look at another class of EDAs, which attempt to learn variable dependencies. Here, we consider the MIMIC algorithm (see Algorithm 3). The reason behind the inclusion of the MIMIC is that this algorithm builds a chain-structured model in each generation using entropy and conditional entropy between decision variables. In order to optimise DLB, algorithms need to correct all blocks from left to right and maintain these blocks over

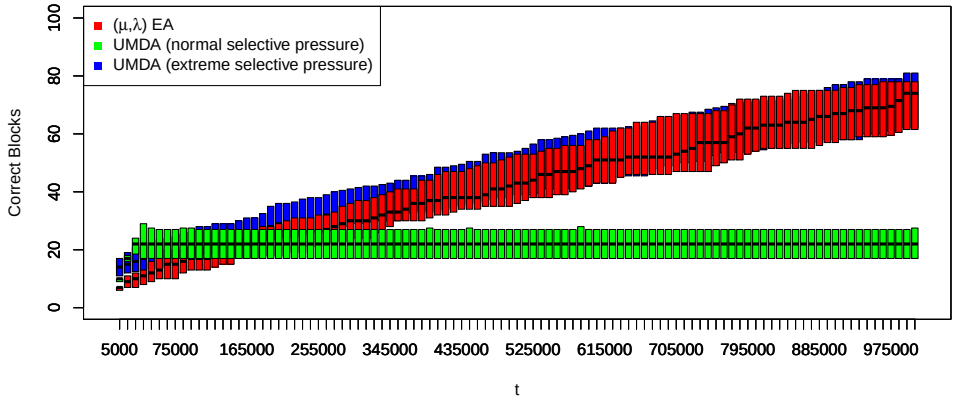
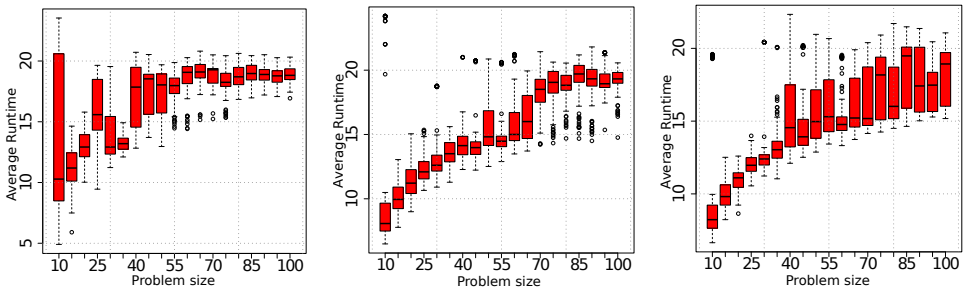


Figure 1: Number of correct blocks in the fittest individuals of the (μ, λ) EA and UMDA on the DLB problem with problem size $n = 200$ as a function of number of function evaluations t. UMDA with extreme selective pressure uses parent population size $\mu = 10$, while the other experiments use $\mu = 200$, $\lambda = 1000$, and mutation rate $1/n$ for the (μ, λ) EA.

generations. Our conjecture is that the MIMIC can deal with small traps in DLB easily. This is because, if the μ-th individual in the sorted population has i leading 11s, then bits at positions $1, 2, \ldots, 2i$ will have the minimum entropy and conditional entropy, which might form the prefix of the chain-structured model. The algorithm might then be able to choose the 'right' bits (i.e., bits $2i + 1$ and $2i + 2$) to add to the prefix and easily increase the value of i in the next generation.

We consider three different settings for the population: $\lambda = \sqrt{n}$ (i.e. small), $\lambda = \sqrt{n} \log n$ (medium) and $\lambda = n$ (large) for $n \in \{10, 15, 20, \ldots, 100\}$. As mentioned before, the MIMIC includes a vast number of probability calculations (to obtain entropy and conditional entropy), but this does not matter as we are only interested in the number of fitness evaluations the algorithm performs. For each value of n, the algorithms are run 100 times and the average runtime is computed. The results are shown in Figure 2.

In general, the results show that the MIMIC can find the optimum of the DLB problem under all choices of population size. When n is small (especially around 10), there is a very large variance in the number of fitness evaluations. This is because the parent population is so small, the model is easily influenced by the few top individuals, which may result in



(a) $\lambda = \sqrt{n}$ (b) $\lambda = \sqrt{n} \log n$ (c) $\lambda = n$

Figure 2: Average runtime of the MIMIC on the DLB function. The y-values are the base-2 logarithm of the number of fitness evaluations. There are 100 runtime measures for each value of the problem instance size n.

a random walk. When the population size becomes larger, the diversity in the population can be maintained for a longer period of time, and the empirical marginals cannot deviate too far from the true marginals; thus, the algorithm looks more stable. Furthermore, the size of the population seems to have a real impact on the runtime of the algorithm. For small and medium size (i.e. only differ by a logarithmic factor), the runtime looks similar; however, for large $\lambda = n$ the MIMIC on average requires a smaller number of fitness evaluations but exhibits a larger variance. Although we cannot provide a rigorous analysis of the MIMIC on the DLB problem, the shapes in all figures suggest that the number of fitness evaluations might be polynomial in problem size n.

6 Conclusions

In this paper, we have introduced the Deceptive Leading Blocks, in which bits are highly correlated (similar to LeadingOnes) and contains many small traps (like a trap function). Since this function is new, we first show that simple EAs can optimise the function in polynomial time. More specifically, we consider three typical EAs, that are $(1 + \lambda)$ EA, $(\mu + 1)$ EA and the non-elitist (μ, λ) EA, and the upper bounds derived verified our claims. Next, we aim at showing that due to correlation and deception, the DLB problem may be hard for the UMDA, which assumes independence between variables.

We are able to show a lower bound of $e^{\Omega(\mu)}$ on the expected runtime of the UMDA on DLB for any selective pressures $\frac{\mu}{\lambda} > \frac{14}{1000}$ and parent population size $\mu = \Omega(\log n)$ and $\mu = o(n)$. On the other hand, if the selective pressure is extremely high (i.e., $\mathcal{O}(1/\mu)$), the UMDA optimises the DLB function in an $\mathcal{O}\left(n\lambda \log \lambda + n^3\right)$ expected runtime.

We believe that the difficulty that the UMDA faces when optimising DLB stems from the first-order statistics underlying the algorithm, that causes the algorithm to forget everything it has learned so far. Motivated by this observation, we look at the class of bivariate EDAs, and a typical one is the MIMIC. Due to the complexity of how the entropy and pairwise conditional entropy measured to build a chain-structured model, we were unable to analyse it using rigorous arguments. Thus, we present some experimental results of the MIMIC on DLB to draw future attention from the community. The experimental findings suggest that the MIMIC might be able to optimise DLB in polynomial runtime and exhibit the ability to fill in the gaps left by the UMDA.

We leave it as an open problem for future work to analyse the behaviour of the algorithm in the case $\frac{14}{1000} > \frac{\mu}{\lambda} = \omega(1/\mu)$. Future work should also consider theoretical aspects of the MIMIC on simple toy functions like OneMax and LeadingOnes. Rigorous arguments should be formed to describe how the chain-structured model is built and how the next population is sampled using pairwise conditional probabilities following the order of bits in the model.

References

[1] D. H. Ackley. An empirical study of bit vector function optimisation. *Genetic Algorithms and Simulated Annealing*, pages 170–204, 1987.

[2] R. Armañanzas, I. Inza, R. Santana, Y. Saeys, J. L. Flores, J. A. Lozano, Y. V. D. Peer, R. Blanco, V. Robles, C. Bielza, and P. Larrañaga. A review of estimation of distribution algorithms in bioinformatics. *BioData Mining*, 1(1):6, 2008.

[3] S. Baluja. Population-based incremental learning: A method for in-

tegrating genetic search based function optimization and competitive learning. *Technical report, Carnegie Mellon University*, 1994.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, 2006.

[5] T. Chen, P. K. Lehre, K. Tang, and X. Yao. When is an estimation of distribution algorithm better than an evolutionary algorithm? In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2009*, pages 1470–1477, 2009.

[6] D. Corus, D.-C. Dang, A. V. Eremeev, and P. K. Lehre. Level-based analysis of genetic algorithms and other search processes. *IEEE Transactions on Evolutionary Computation*, 22(5):707–719, Oct 2018.

[7] D. C. Dang and P. K. Lehre. Simplified runtime analysis of estimation of distribution algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '15, pages 513–518, 2015.

[8] D. C. Dang, P. K. Lehre, and P. T. H. Nguyen. Level-based analysis of the univariate marginal distribution algorithm. *Algorithmica*, 2018.

[9] Y. Davidor. Epistasis variance: A viewpoint on GA-hardness. volume 1 of *Foundations of Genetic Algorithms*, pages 23 – 35. Elsevier, 1991.

[10] J. S. De Bonet, C. L. Isbell Jr, and P. Viola. MIMIC: Finding optima by estimating probability densities. In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, NIPS '96, pages 424–430, 1996.

[11] B. Doerr. A tight runtime analysis for the cga on jump functions - edas can cross fitness valleys at no extra cost. *CoRR*, abs/1903.10983, 2019.

[12] B. Doerr. An exponential lower bound for the runtime of the cga on jump functions. *CoRR*, abs/1904.08415, 2019.

[13] B. Doerr and C. Doerr. The impact of random initialization on the runtime of randomized search heuristics. *Algorithmica*, 75(3):529–553, 2016.

[14] B. Doerr and T. Kötzing. Multiplicative up-drift. *CoRR*, abs/1904.05682, 2019.

[15] B. Doerr and M. S. Krejca. Significance-based estimation-of-distribution algorithms. In *Proceedings of Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1483–1490, 2018.

[16] B. Doerr, D. Johannsen, and C. Winzen. Multiplicative drift analysis. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '10, pages 1449–1456, 2010.

[17] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+ 1) evolutionary algorithm. *Theoretical Computer Science*, 276(1-2):51–81, 2002.

[18] D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 1st edition, 2009.

[19] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. SpringerVerlag, 2003.

[20] B. Eisenberg. On the expectation of the maximum of iid geometric random variables. *Statistics & Probability Letters*, 78(2):135 – 143, 2008.

[21] W. Feller. *An introduction to probability theory and its applications*, volume 1. Wiley, 3 edition, 1968.

[22] T. Friedrich, T. Kötzing, and M. S. Krejca. EDAs cannot be balanced and stable. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '16, pages 1139–1146, 2016.

[23] T. Friedrich, T. Kötzing, M. S. Krejca, and A. M. Sutton. The compact genetic algorithm is efficient under extreme gaussian noise. *IEEE Transactions on Evolutionary Computation*, 21(3):477–490, 2017.

[24] R. Gras. How efficient are genetic algorithms to solve high epistasis deceptive problems? In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 242–249, 2008.

[25] G. R. Harik, F. G. Lobo, and D. E. Goldberg. The compact genetic algorithm. *IEEE Trans. Evolutionary Computation*, 3(4):287–297, 1999.

[26] V. Hasenöhrl and A. M. Sutton. On the runtime dynamics of the compact genetic algorithm on jump functions. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 967–974, 2018.

[27] M. Hauschild and M. Pelikan. An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation*, 1(3): 111–128, 2011.

[28] J. He and X. Yao. Towards an analytic framework for analysing the computation time of evolutionary algorithms. *Artificial Intelligence*, 145 (1):59 – 97, 2003.

[29] T. Jansen and R. P. Wiegand. The cooperative coevolutionary (1+1) EA. *Evolutionary Computation*, 12(4):405–434, 2004.

[30] T. Jansen, K. A. De Jong, and I. Wegener. On the choice of the offspring population size in evolutionary algorithms. *Evolutionary Computation*, 13(4):413–440, 2005. ISSN 1063-6560.

[31] D. Johannsen. *Random combinatorial structures and randomized search heuristics*. PhD thesis, Universität des Saarlandes, Germany, 2010.

[32] M. S. Krejca and C. Witt. Lower bounds on the run time of the univariate marginal distribution algorithm on onemax. In *Proceedings of the 14th ACM/SIGEVO Conference on Foundations of Genetic Algorithms*, FOGA '17, pages 65–79, 2017.

[33] P. Larrañaga and J. A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Genetic Algorithms and Evolutionary Computation. Springer US, 2001.

[34] P. K. Lehre. Negative drift in populations. In *Parallel Problem Solving from Nature, PPSN XI*, pages 244–253. Springer Berlin Heidelberg, 2010.

[35] P. K. Lehre. Fitness-levels for non-elitist populations. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '11, pages 2075–2082, 2011.

[36] P. K. Lehre and P. T. H. Nguyen. Improved runtime bounds for the univariate marginal distribution algorithm via anti-concentration. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, pages 1383–1390, 2017.

[37] P. K. Lehre and P. T. H. Nguyen. Level-based analysis of the population-based incremental learning algorithm. In *Proceedings of the International Conference on Parallel Problem Solving from Nature*, PPSN XV, pages 105–116, 2018.

[38] P. K. Lehre and P. T. H. Nguyen. Runtime analysis of the univariate marginal distribution algorithm under low selective pressure and prior noise. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '19, 2019. To appear.

[39] P. K. Lehre and P. S. Oliveto. *Theoretical Analysis of Stochastic Search Algorithms*, pages 1–36. Springer International Publishing, 2018.

[40] J. Lengler, D. Sudholt, and C. Witt. Medium step sizes are harmful for the compact genetic algorithm. In *Proceedings of Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1499–1506, 2018.

[41] R. Motwani and P. Raghavan. *Randomised algorithms*. Cambridge University Press, 1995.

[42] H. Mühlenbein and G. Paaß. *From recombination of genes to the estimation of distributions I. Binary parameters*, pages 178–187. 1996.

[43] M. Pelikan, D. E. Goldberg, and F. G. Lobo. A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications*, 21(1):5–20, 2002.

[44] D. Simon. *Evolutionary Optimisation Algorithms*. Wiley, 2013.

[45] M. Slatkin. Linkage disequilibrium — understanding the evolutionary past and mapping the medical future. *Nature Reviews Genetics*, 9(6): 477–485, 2008.

[46] D. Sudholt. How crossover speeds up building block assembly in genetic algorithms. *Evolutionary Computation*, 25(2):237–274, 2017.

[47] D. Sudholt and C. Witt. Update strength in edas and aco: How to avoid genetic drift. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, pages 61–68, 2016.

[48] I. Wegener. *Methods for the Analysis of Evolutionary Algorithms on Pseudo-Boolean Functions*, pages 349–369. 2002.

[49] E. W. Weisstein. Binomial distribution. URL `http://mathworld.wolfram.com/BinomialDistribution.html`.

[50] C. Witt. Runtime analysis of the (μ+1) EA on simple pseudo-boolean functions. *Evolutionary Computation*, 14(1):65–86, 2006.

[51] C. Witt. Upper bounds on the runtime of the univariate marginal distribution algorithm on onemax. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, pages 1415–1422, 2017.

[52] C. Witt. Domino convergence: why one should hill-climb on linear functions. In *Proceedings of Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1539–1546, 2018.

[53] Z. Wu, M. Kolonko, and R. H. Möhring. Stochastic runtime analysis of a cross entropy algorithm. *IEEE Transactions on Evolutionary Computation*, 21(4):616–628, 2017.

[54] W. Zheng, G. Yang, and B. Doerr. Working principles of binary differential evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1103–1110, 2018.